INTELLIGENT REAL-TIME DECISION SUPPORT SYSTEMS FOR ROAD TRAFFIC MANAGEMENT

Multi-agent based Fuzzy Neural Networks
with a GA learning approach in managing
control actions of road traffic centres

Khaled A. ALMEJALLI MSc

Submitted for the degree
of Doctor of Philosophy

Department of Computing

University of Bradford

2010

Abstract

Khaled Almejalli

Intelligent Real-Time Decision Support Systems

for Road Traffic Management

Keywords: road traffic management and control, decision support system, fuzzy rule identification,fuzzy neural network, genetic algorithm, multi-agent systems

The selection of the most appropriate traffic control actions to solve non-recurrent traffic congestion is a complex task which requires significant expert knowledge and experience. In this thesis we develop and investigate the application of an intelligent traffic control decision support system for road traffic management to assist the human operator to identify the most suitable control actions in order to deal with non-recurrent and non-predictable traffic congestion in a real-time situation. Our intelligent system employs a Fuzzy Neural Networks (FNN) Tool that combines the capabilities of fuzzy reasoning in measuring imprecise and dynamic factors and the capabilities of neural networks in terms of learning processes. In this work we present an effective learning approach with regard to the FNN-Tool, which consists of three stages: initializing the membership functions of both input and output variables by determining their centres and widths using self-organizing algorithms; employing an evolutionary Genetic Algorithm (GA) based learning method to identify the fuzzy rules; tune the derived structure and parameters using the back-propagation learning algorithm. We evaluate experimentally the performance and the prediction capability of this three-stage learning approach using well-known benchmark examples. Experimental results demonstrate the ability of the learning approach to identify all relevant fuzzy rules from the training data. A comparative analysis shows that the proposed learning approach has a higher degree of predictive capability than existing models. We also address the scalability issue of our intelligent traffic control decision support system by using a multi-agent based approach. The large network is divided into sub-networks, each of which has its own associated agent. Finally, our intelligent traffic control decision support system is applied to a number of road traffic case studies using the traffic network in Riyadh, in Saudi Arabia. The results obtained are promising and show that our intelligent traffic control decision support system can provide an effective support for real-time traffic control.

Acknowledgement

This thesis would not have been possible without the guidance of my supervisors and support from my family.

Most of all, I would like to thank God, ALLAH, for having made everything possible by giving me strength and courage to do this work.

My deepest gratitude to my supervisors Dr. Keshav Dahal and Dr. Alamgir Hossain for not only giving me the opportunity to work with them but also for their constant encouragement, guidance, support and patience they demonstrated throughout my PhD research. Our frequent meeting and discussions have been extremely useful and stimulating; your comments and suggestions have helped me to follow the right direction in my work. Thank you very much for your advice, and support.

I am really indebted to the members of MOSAIC (Modelling Optimisation Scheduling And Intelligent Control) research group for their collaboration and supporting me during my PhD research.

Last, but not least, I would like to express my heartfelt thanks to my mother for her prayers and moral support during my study. I am also deeply indebted to my wife for supporting me during the whole time, putting up with the long hours I spent with my study, and for taking care of our kids while I was working.

Glossary

List of symbols

ca_i	control action *i*.
P^i	aggregated performance of control action ca_i.
C_d	performance criterion.
C_d^{min}	minimum values of C_d.
C_d^{max}	maximum values of C_d.
w_{C_d}	weight of the performance criterion C_d.
$E_{C_d}^i$	evaluation of control action ca_i over the performance criterion C_d.
$o_n^{(1)}$	output of node *n* at L*ayer 1 of* FNN-Tool.
$IL_{n,m}^{(2)}$	*m*th input label of the linguistic node *n*.
$o_{n,m}^{(2)}$	output of input-label node $IL_{n,m}^{(2)}$ at L*ayer 2 of* FNN-Tool.
$c_{n,m}^{(2)}$	centres of the membership function for the input-label node $IL_{n,m}^{(2)}$.
$\sigma_{n,m}^{(2)}$	widths of the membership function for the input-label node $IL_{n,m}^{(2)}$.
$o_u^{(3)}$	output of rule node RL_u at L*ayer 3 of* FNN-Tool.
$o_{n,m}^{(4)}$	output of consequence node $OL_{n,m}$ at Layer 4 of FNN-Tool.
$W_{u,nm}$	connection weights of the links connecting nodes RL_u at Layer 3 to $OL_{n,m}$ at layer 4.
y_n	output of an output node D_n at Layer 5 of FNN-Tool.
$c_{u,nm}^{(4)}$	centre of the membership function of the output label node $OL_{n,m}$.

$\sigma_{u,nm}^{(4)}$	width of the membership function of the output label node $OL_{n,m}$.
$e_n^{(i)}$	error signal in Layer i, $i \in \{2,3,4,5\}$.
ca_i	control action i
CA_{table}	control actions data-table
sn_j	affected sub-networks
g_j	identity (or name) of the agent that controls the sub-network (sn_j).
Y_j^i	percentage change (positive or negative) that may happen in the traffic flows to the affected sub-network (sn_j) due to the application of the traffic control action ca_i .
R_j^i	outflow restrictions of affected sub-network (sn_j) due to the application of the traffic control action ca_i .
$P\_Dem_j^i$	predicted traffic demand of the sub-network associated with agent g_j due to the application of the traffic control action ca_1 .
C_Dem_j	current traffic demand of the sub-network associated with agent g_j .
F_j^i	fitness of ca_i which is provided by the affected agent g_j .
p_l^i	local fitness of the control action ca_i which is provided by agent *A*.
p_g^i	global performance of the control action ca_i .
ω_j	weights which represents the relative importance of the affected agent g_j .
μ_j^i	measure that shows how much impact the traffic control action ca_i is having on the affected sub-network associated with agent g_j .
Max_OF_j	maximum possible traffic inflow into the affected sub-network associated with agent g_j.

Acronyms and Abbreviations

ANN	Artificial Neural Network.
GA	Genetic Algorithm.
FNN	Fuzzy Neural Network.
VMS	Variable Message Signs.
VSL	Variable Speed Limit
ITC-DSS	Intelligent Traffic Control Decision Support System.
FNN-Tool	Fuzzy Neural Network Tool.
TTT	Total Travel Time.
TDT	Total Distance Travelled .
TDm	average Traffic Demand.
TDn	average Traffic Density.
OFR	Outflow Restrictions .
IS	Incidents Status.
RMS	Root Mean Square .
BP	Back-Propagation.

Contents

Chapter 1 Introduction 1

1.1 Background and Motivation 1

1.2 Aim and Objectives of the Research 4

1.3 Contribution of the Thesis 6

1.4 Deliverables 7

1.5 Thesis Outline 10

Chapter 2 Overview of Road Traffic Management and Control 12

2.1 Chapter Overview 12

2.2 Traffic Measurements, Sensors and Control 13

2.2.1 Traffic Variables 13

2.2.2 Sensor Systems 15

2.2.3 Traffic Control Actions 18

2.3 Traffic Flow Modelling 23

2.4 METANET 25

2.5 Summary 27

Chapter 3 Literature Review 28

3.1 Chapter Overview 28

3.2 Background to AI Techniques 29

3.2.1 Fuzzy Logic 29

3.2.2 Artificial Neural Network 30

3.2.3 Genetic Algorithms 32

3.2.4 Intelligent Multi-Agent 34

3.3 A Survey of Intelligent Traffic Management Systems 35

3.3.1 Traffic Problem Detection Systems 35

3.3.2 Traffic Simulation and Prediction Systems 39

3.3.3 Traffic Control Systems 42

3.4 Summary 49

Chapter 4 An Intelligent Traffic Control Decision Support System 51

4.1 Chapter Overview 51

4.2 Overview Fuzzy Neural Networks 52

4.3 The Proposed Decision Support System 54

4.3.1 Overall System Framework 54

4.3.2 Aggregated Performance 56

4.3.3 Fuzzy Neural Network Tool (FNN-Tool) Structure 59

4.4 Application of ITC-DSS 63

4.4.1 Riyadh Traffic Case Study 64

4.4.2 Case Study Design and Testing 65

4.5 Summary 68

Chapter 5 GA-Based Fuzzy Rules Identification Method for Fuzzy Neural Networks 70

5.1 Chapter Overview 70

5.2 Fuzzy-Rule Identification 71

5.2.1 Rule-identification Methods 71

5.2.2 Genetic Fuzzy System 74

5.3 The Proposed GA-based Fuzzy Rule Generating Algorithm 76

5.3.1 GA-based Method 76

5.3.2 GA Design 79

5.4 Validation Experiment 82

5.5 Summary 84

Chapter 6 Learning Approach for Fuzzy Neural Networks 86

6.1 Chapter Overview 86

6.2 FNN Learning Methods 87

6.3 The Proposed Learning Approach 92

6.3.1 Stage 1: Initialization with a Self- Organizing Algorithm 92

6.3.2 Stage 2: Rule Identification with GA-Based Method 95

6.3.3 Stage 3: Fine Tuning Stage with a Back-Propagation Learning Algorithm 95

6.4 Experimental Results and Analysis 99

6.4.1 Nonlinear System Identification Example 100

6.4.2 Experimental Design 102

6.4.3 GA Operations and Parameters 105

6.4.4 Experimental Results 108

6.4.5 Comparison with Other Models 113

6.5 Road Traffic Case Study 115

6.5.1 Traffic Data Generation 116

6.5.2 Learning FNN-Tool 120

6.5.3 Result Validation 126

6.6 Summary 127

Chapter 7 A Multi-Agent Approach for Intelligent Traffic Control Systems 129

7.1 Chapter Overview 129

7.2 Global Network-Oriented Traffic Control 130

7.3 The Proposed Multi-agent System 132

7.3.1 Structure 133

7.3.2 Operation 136

7.3.3 Calculation of Control Action Fitness 138

7.3.4 Global Performance of Control Action 139

7.4 Case Studies and Experimental Results 141

7.4.1 Riyadh Traffic Case Study 141

7.4.2 Application of the Proposed Multi-agent Approach 143

7.4.3 Different Traffic Scenarios 151

7.4.4 Cross-Scenario Analysis 167

7.5 Summary 172

Chapter 8 Conclusions and Future Work 174

8.1 Conclusions 174

8.2 Usability Issues 178

8.3 Future Work 182

List of Table

Table 4.1: The performance evaluation of the control actions $ca1, ca2, ca3, ca4$, and $ca5$ on the selected traffic state. 67

Table 6.1: Fuzzy rules generated from the second and the third stages of the learning process and their corresponding weights of the Box-Jenkins gas furnace data. 109

Table 6.2: Comparative results for different modelling approaches........114

Table 6.3: Maximum, minimum, and average value of data set for the considered traffic case study. 119

Table 6.4: Initial and final parameters of the MFs for all variables of the traffic case study. 121

Table 6.5: $R2$ of the TTT and TDT for the five control actions 127

Table 7.1: The structure of the proposed control actions data table $CAtable$ 135

Table 7.2: The considered traffic states of the three sub-networks. 144

Table 7.3: The current and the maximum possible traffic demands ($C_Demj$and Max_OFj) of Olaya and Takhassusi sub-networks coming from King Fahad sub-network. 145

Table 7.4: $CAtable$ for the King Fahd sub-network with the five control actions. 146

Table 7.5: the values of wCd , $Cdmin$ and $Cdmax$ used by agents B and C to calculate the aggregated performance of each control action 147

Table7.6: The predicted performance of the five control actions for agent B. 148

Table 7.7: The predicted performance of the five control actions for agent C. 149

Table 7.8: Summary of the final results of the proposed multiagent approach 150

Table 7.9: Brief description of the ten control actions considered in this part of the experiment. 152

Table 7.10: The $CAtable$ for the King Fahad sub-network with the ten control actions. 153

Table 7.11: Summary of the final results of the proposed multi-agent system for the first scenario. 158

Table 7.12: Summary of the final results of the proposed multi-agent system for the second scenario. 160

Table 7.13: Summary of the final results of the proposed multi-agent system for the third scenario. 162

Table 7.14: Summary of the final results of the proposed multi-agent system for the fourth scenario. 164

Table 7.15: Summary of the final results of the proposed multi-agent system for the fifth scenario. 166

List of Figures

Figure 1.1: Typical information infrastructure for real-time traffic control. 2

Figure 2.1: Inductive Loop Detector System. 16

Figure 2.2: An example of a video image of a highway taken by a traffic detector camera..17

Figure 2.3: The installation of a mobile pneumatic tube traffic detector. The tube is fixed on the highway surface and connected to a registration unit at the roadside...18

Figure 2.4: An example of ramp metering system (freeway-to-freeway metering). . 19

Figure 2.5: An example of variable speed limits sign setup on a highway. 20

Figure 2.6: An example of variable messages signs setup on a highway. 21

Figure 2.7: Green arrows and red crosses on VSL show which lanes are open and which are closed. 22

Figure 2.8: A section of a highway modelled by METANET. 26

Figure 4.1: Structure of Intelligent Traffic Control Decision Support System (ITC-DSS). 55

Figure 4.2: The main process of the proposed ITC-DSS 56

Figure 4.3: Structure of the Fuzzy Neural Network-Tool (FNN-Tool). 59

Figure 4.4: Gaussian membership function. 61

Figure 4.5: The sub-network considered in the traffic case study. 64

Figure 5.1: (a) All possible rules with two inputs and one output; (b) an example of encoding a chromosome. 78

Figure 5.2: The working principle of a simple genetic algorithm. 81

Figure 5.3: Considered GA operations (a) two point crossover; (b) mutation. 82

Figure 5.4 Performance graph for 100 generation of 20 chromosomes...................84

Figure 6.1: Flowchart of the proposed three stage-based learning approach............91

Figure 6.2: The characterization of the Box-Jenkins gas furnace data: original time data. 100

Figure 6.3: The characterization of the Box-Jenkins gas furnace data: auto correlation function. 101

Figure 6.4: The characterization of the Box-Jenkins gas furnace data: Histogram. 101

Figure 6.5: The initialized FNN-Tool Structure for the Box-Jenkins gas furnace data 103

Figure 6.6: The initial membership functions generated from the first stage of the learning process of FNN-Tool...104

Figure 6.7: GA average performance of 10 runs using a population size of 90, two-point crossovers with probability of 0.7, and a mutation probability of 0.05. 106

Figure 6.8: Sensitivity to crossover type and probability using mutation probability of 0.05...107

Figure 6.9: Sensitivity to mutation probability using a two-point crossover with a probability of 0.7. 107

Figure 6.10: Sensitivity to population size using a two-point crossover with probability of 0.7 and a mutation probability of 0.05............................108

Figure 6.11: The improvement of the FNN-Tool prediction error of the Box-Jenkins gas furnace data after the BP algorithm. 110

Figure 6.12: The learned input-output membership functions of the Box-Jenkins gas furnace data. 111

Figure 6.13: The FNN-Tool performance of the Box-Jenkins gas furnace data: (a) desired and predicted output; (b) training and testing distribution; (c) the GAFNN prediction error. 112

Figure 6.14: The considered traffic network drawn via METANET......................118

Figure 6.15: the initialized FNN-Tool Structure for the traffic case study..............120

Figure 6.16: The FNN-Tool prediction error of the traffic case study after the second stages of the learning process..122

Figure 6.17: The improvement of the error rate after the BP algorithm.................123

Figure 6.18: (a) Desired and predicted output *TTT*; (b) Desired and predicted output *TDT*..124

Figure 6.19: (a) The FNN-Tool prediction error of *TTT*; (d) the FNN-Tool prediction error of *TDT*...125

Figure 7.1: Centralized and decentralized coordination in multi-agent systems.....132

Figure 7.2: The overall structure of the proposed multi-agent system. 134

Figure 7.3: Flowchart of the process of the proposed multi-agent system. 137

Figure 7.4: The Riyadh traffic network considered in the case study......................143

Figure 7.5: The first traffic scenario for the three traffic sub-networks. 155

Figure 7.6: The second traffic scenario for the three traffic sub-networks.............155

Figure 7.7 The third traffic scenario for the three traffic sub-networks. 156

Figure 7.8: The fourth traffic scenario for the three traffic sub-networks..............156

Figure 7.9: The fifth traffic scenario for the three traffic sub-networks..................157

Figure 7.10: The local and global performance of the ten control actions for the first scenario. 159

Figure 7.11: The local and global performance of the ten control actions for the second scenario. 161

Figure 7.12: The local and global performance of the ten control actions for the third scenario. 163

Figure 7.13: The local and global performance of the ten control actions for the fourth scenario. 165

Figure 7.14: The local and global performance of the ten control actions for the fifth scenario. 167

Chapter 1

Introduction

1.1 Background and Motivation

The traffic congestion problem has become serious as the number of vehicles on our roads and the need for transportation grows. Traffic congestion not only causes considerable costs due to unproductive time losses, it also increases the probability of accidents as well as having a negative impact on the environment (e.g. air pollution, lost fuel) and on the quality of life (e.g. health problems, noise, stress) [12]. Therefore, traffic management and control have become a major problem in developing as well as developed countries. Governments have been spending hefty amounts to develop traffic control centres using different methodologies, by incorporating the benefits of advanced information technology.

Modern traffic control centres are connected to monitoring devices such as detectors, weather sensors, and cameras, to record data related to the traffic state online, e.g. speed (km/h), flow (vehicle/h), occupancy (percentage of time the sensor is occupied by a vehicle), environmental conditions (such as air and ground temperature, amount of precipitation, wind velocity and direction, etc.). Moreover, the control centres use advanced dynamic control devices such as traffic signals at intersections, traffic signals at on-ramps, variable message signs (VMSs) that can present different messages to motorists (e.g., warning about an existing congestion, alternative path, or weather warning), radio advisory systems to broadcast messages to motorists, etc. Figure 1 shows a typical infrastructure for real-time traffic control that can be found in different cities [12].

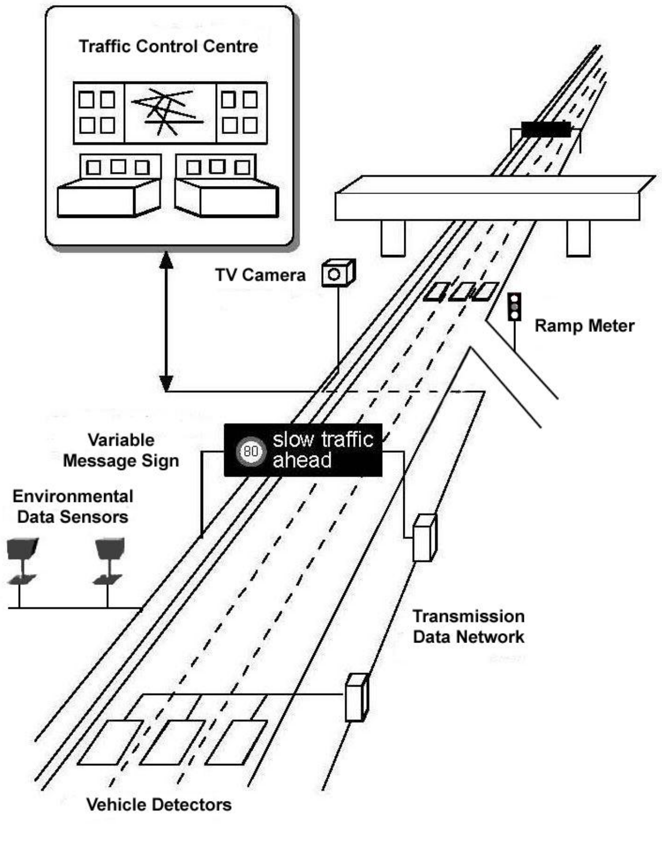


Figure 1.1: Typical information infrastructure for real-time traffic control [12].

When non-recurrent congestion happens, the operator at the traffic control centre has to assess the severity of the situation, predict the most probable evolution of the state of the network, and select the most appropriate control actions quickly [57]. There are a large number of factors related to the state of traffic and a large number of possible control actions that need to be considered during the decision making process. Also the operator should consider the interrelations between traffic situations at different locations in the network and the interrelations between the traffic control actions at different locations in the network. The identification of suitable control actions for a given non-recurrent traffic congestion situation can be tough, even for experienced operators. Therefore, an advanced traffic control system that integrates the traffic state data with traffic monitoring and control software to help operators in decisions making, is needed. Road traffic simulation models are used in many cases. However, simulating different traffic scenarios for a number of control measures in a complicated situation can be very time-consuming [30].

The concept “intelligent” is very popular among road traffic management and control applications. In these cases the intelligence refers to the capability of a system to adapt when surrounding conditions change and its capability to learn from data to provide assistance to decision makers in road traffic management. The intelligent techniques including (knowledge-based, case-based reasoning, fuzzy logic, artificial neural networks, genetic algorithms and intelligent agents) have been widely applied to traffic management applications in the last decade [12]. Also the increase in performance of computers has improved the applicability of these intelligent techniques.

There are several reasons behind the wide use of intelligent techniques in integrated road traffic management applications. Kirschfink et al. [91] offered three main reasons for this. Firstly, the current traffic management and control systems in most large-sized urban networks show limitations when it comes to facing critical traffic problems. Secondly, in most traffic management centres, even though advanced traffic control technologies are used, human operators still play a critical role in day-to-day operations. Thirdly, the recent significant development of traffic monitoring, incident detection and management facilities has increased the need for intelligent on-line tools to help traffic management operators to cope with the complexity of both the information managed and of the resulting, integrated traffic management schemes. In this context, the development of intelligent systems that are capable of managing traffic behaviour and evolution in similar ways to that of an expert traffic operator is required.

Another motivation for the investigation of intelligent traffic control systems comes from the fact that the PhD researcher worked in the Riyadh Region Traffic Department, which controls the very complex road network serving the capital of Saudi Arabia. He has had various experiences and responsibilities concerning road traffic management. He has seen how an effective and safe control of the road network helps to prevent traffic congestion and traffic accidents, and how negligence can cause loss of life. He decided to pursue this topic because, he hoped, it could help improve road traffic management and promote better safety for drivers and pedestrians.

1.2 Aim and Objectives of the Research

The overall aim of this research is to investigate the development of a real-time intelligent traffic control decision support system for traffic control centres. This intelligent decision support system should help the human traffic operator to identify the most suitable control actions to deal with non-recurrent, non-predictable traffic congestion. The system is not designed to replace the human traffic operator but it should act as intelligent tool that assists the operator to react in a uniform and structured way to unusual situations.

The intelligent traffic control decision support system should have the following elements:

- **Prediction:** it should be able to accurately predict the performance of several control actions for a given traffic state.
- **Performance evaluation:** it should be able to evaluate the performance of the control actions based on user-specified objectives.
- **Optimization:** it should be able to find the control action that optimizes the user's objectives.
- **Processing speed:** it should run in real time.

The main objectives with regard to achieving the aim of this research are as follows:

- To explore the literature on road traffic management concepts (including road traffic variables, traffic sensor technologies, traffic control actions and road traffic flow modelling) and intelligent road traffic control systems.
- To explore the merits of artificial intelligent techniques. This includes fuzzy logic, neural networks, multi-agent, and generic algorithms.
- To develop an effective real-time traffic control decision support system using intelligent techniques for identifying the most suitable control action in real-time.

- To develop an effective learning algorithm for the developed control decision support system.
- To develop an effective method/framework to scale-up the developed real-time traffic control decision support system to be used in a large traffic network.
- To demonstrate the effectiveness of the developed system by applying it to several case studies.

1.3 Contribution of the Thesis

The research contributions presented in this thesis can by summarized as follows:

1. Extensive literature review on road traffic management, controls, and systems.
2. Intelligent traffic control decision support system (ITC-DSS): The proposed intelligent system has the potential to be used for assisting the human operator to manage the current traffic state in real-time. The developed system should allow the operator to quickly evaluate a set of traffic control actions in one process, unlike the process of evaluating these actions one by one, as in the simulation model. The novelty of the developed traffic control decision support system primarily lies in employing an adaptive fuzzy neural network which combines the capability of fuzzy reasoning in handling uncertain information and the capability of neural networks in learning from processes.

3. GA based fuzzy rules identification method for fuzzy neural networks: The main contribution of the developed GA based method is the direct encoding of the fuzzy rules in GA chromosomes using integer representation rather than the conventional encoding of the weights of the fuzzy rules. This reduces the length of the chromosome as well as making the size of the GA search space small.

4. Intelligent three stage-based learning approach for fuzzy neural networks: The main distinction of the developed learning approach is that the process of determining the relevant fuzzy rules and the process of fine tuning rule weights are implemented in two separate learning stages.

5. Development of a multi-agent framework for controlling a large traffic network. A new method for predicting the global performance of traffic control actions has been employed in the developed multi-agent framework. The simplicity and manageability represent the novelty of the developed multi-agent framework.

1.4 Deliverables

In addition to the material presented in this thesis, the findings of this research have been disseminated through a book chapter, journals, conferences, workshops and colloquiums.

Submitted to Journals

1. K. Almejalli, K. Dahal, and A. Hossain, " An Intelligent Multi-Agent Traffic

Control System," Submitted to IEEE Transactions on Intelligent Transportation Systems.

2. K. Almejalli, K. Dahal, and A. Hossain, " Evolutionary Approach for Training Fuzzy Neural Networks in Absence of Expert Knowledge" Submitted to IEEE Transactions on Fuzzy Systems.

Chapter in Books

1. K. Almejalli, K. Dahal, and A. Hossain, "Real Time Identification of Road Traffic Control Measures," in Advances in Computational Intelligence in Transport, Logistics, and Supply Chain Management, A. Fink and F. Rothlauf, Eds. Springer, 2008, ch. 4, pp. 63-80.

Refereed Conferences

1. K. Almejalli, K. Dahal, and A. Hossain, "An Intelligent Multi-agent Approach for Road Traffic Management Systems," in Proceedings of the 18th IEEE International Conference on Control Applications (CCA), Saint Petersburg, RUSSIA, 2009.

2. K. Almejalli, K. Dahal, and A. Hossain, "Intelligent Traffic Control Decision Support System," in Proceedings of European Workshop on Evolutionary Computation in Transportation and Logistics, EvoTransLog 2007, LNCS Springer-Verlag, vol. 4448, pp. 688-701, 2007.

3. K. Almejalli, K. Dahal, and A. Hossain, "GA-Based Learning Algorithms to Identify Fuzzy Rules for Fuzzy Neural Networks," in Proceedings of The 7th International Conference on Intelligent Systems Design and

Applications, IEEE Computer Science, ISDA2007, Rio de Janeiro, Brazil, 2007, pp. 289-296.

4. K. Almejalli, K. Dahal, and A. Hossain, "Road Traffic Decision Support System," in Proceedings of International Conference on Software Knowledge Information Management and Applications (SKIMA), Chiang Mai, Thailand, 2006, pp. 85-89.

Workshop and Colloquium Papers

1. K. Almejalli, K. Dahal, and A. Hossain, "Intelligent Predictive Decision Support System," in Proceedings of the Saudi Innovation Conference (SiC), Leeds, 2008.
2. K. Almejalli, K. Dahal, and A. Hossain, "A Multi-agent Framework for Intelligent Traffic Management Systems," in Proceedings of the 9th Informatics Research Workshop, University of Bradford, Bradford, 2008, pp. 60-63.
3. K. Almejalli, K. Dahal, and A. Hossain, "Identifying Fuzzy Rules for Fuzzy Neural Networks Using GA-Based Learning Algorithms," in Proceedings of the 8th Informatics Research Workshop, University of Bradford, Bradford, 2007, pp. 62-65.
4. K. Almejalli, K. Dahal, and A. Hossain, "Road Traffic Decision Support System," in Proceedings of the 7th Informatics Research Workshop, University of Bradford, Bradford, 2006.

1.5 Thesis Outline

The rest of the thesis is organized as follows:

Chapter 2 presents a general review of the main topics related to road traffic management, including a brief introduction of METANET, the road traffic simulation model which has been used in this study.

Chapter 3 provides a brief survey of the existing work in the field of road traffic management and control. This survey contains the related work for modern and intelligent road traffic management techniques.

Chapter 4 introduces an Intelligent Traffic Control Decision Support System (ITC-DSS) which have been developed for road traffic control centres to assist the human operator to manage the current traffic state in real-time. The developed ITC-DSS employs a pre-trained Fuzzy Neural Network tool (FNN-Tool) to produce a ranked list of suitable control actions based on the current traffic state. The framework and process of the ITC-DSS, including the structure and the function of the FNN-Tool, is described in this chapter.

Chapter 5 presents a GA-based method for identifying fuzzy rules from numerical data, using previously defined and fixed fuzzy sets. This chapter also gives a brief review of the rule identification methods in the fuzzy neural network framework.

Chapter 6 introduces a three-stage learning approach for fuzzy neural networks. This learning approach has been developed in order to train the proposed FNN-Tool. This chapter firstly reviews the existing learning methods of fuzzy neural networks and then highlights the three stages of the proposed learning approach. Next, it validates the performance of the FNN-Tool using a well-known benchmark example and shows the strength of the FNN-Tool by comparing it with other existing models. Finally, it discusses the application of the proposed learning approach to a traffic case study.

Chapter 7 presents a multi-agent based approach for road traffic control systems. The system proposed in this chapter is a major extension of ITC-DSS presented in Chapter 4. The chapter initially discusses the need for global network-oriented traffic control, then gives a brief description of the intelligent multi-agent and its architecture in the field of road traffic management. Subsequently, the proposed multi-agent approach is introduced. Finally, the application of the proposed system in a case study is used to demonstrate the capabilities of the proposed system.

Chapter 8 summarizes the main contributions of the thesis, and then proposes further research directions.

Chapter 2

Overview of Road Traffic Management and Control

2.1 Chapter Overview

The main purpose of this study is to investigate the development of an intelligent decision support system for road traffic management. In this chapter a general review of the main related topics of the road traffic management is presented. The aim of this chapter is to give the reader an understanding of the basic concepts associated with road traffic management. The chapter is organized as follows. A brief background on road traffic variables, traffic sensor technologies, and traffic control actions is described in Section 2.2. This is followed by a discussion of different road traffic flow modelling in Section 2.3. Finally, the road traffic simulation model METANET, which will be used in this study, is introduced in Section 2.4.

2.2 Traffic Measurements, Sensors and Control

Recently, road traffic management has developed a significant demand for advanced information technology. Modern traffic control centres are connected with advanced dynamic control devices such as detectors, sensors, and cameras to record data related to the traffic state on-line, e.g. speed (km/h), traffic flow (vehicle/h), occupancy (percentage of time the sensor is occupied by a vehicle), etc. All this information arrives periodically at the control centre (e.g., every minute). Moreover, the control centre also receives information about the current state of the control devices. Control devices include traffic signals at intersections, traffic signals at on-ramps, and/or variable message signs (VMSs) that can present different messages to motorists (e.g., warning about an existing congestion, alternative path, or weather warnings), radio advisory systems to broadcast messages to motorists, and reversible lanes (i.e., freeway lanes whose direction can be selected according to the current and expected traffic demand). This section provides a general description of the most important road traffic variables, traffic sensor technologies, and traffic control actions.

2.2.1 Traffic Variables

When a traffic state on the highway is represented based on traffic measurements, several traffic variables are used for describing such a traffic state. This section gives a brief description of the most important traffic variables.

Traffic Flow (or Volume)

Traffic flow is one of the easiest traffic variables used to quantify traffic intensity on the highway. It is defined as the number of vehicles that pass a fixed location on the highway during a certain time period. Traffic flow is defined as vehicles per time unit. It can be recorded for a given section of a highway daily, hourly or sub-hourly. For example, traffic flow = 100 vehicles/hour means that 100 vehicles pass a given section in one hour. A short period interval such as 15min is recommended to ensure that any short-term peak flow which may occur at sub-hourly interval is recorded. Traffic flow is very useful for highway planning[133].

Mean Speed

The mean speed of vehicles is another important measure to assess the traffic situation on the highway. The mean speed of vehicles is defined as the average speeds of all vehicles passing a certain point (the sensor) on the highway over a specified time period expressed in kilometres per hour (km/h) [72]. Mean speed can be measured (e.g. using radar guns or speed camera) for each lane separately or for all the lanes together.

Traffic Density

Traffic density is defined as the number of vehicles on a given length of highway at a certain time instant. It is expressed in vehicles per kilometres per lane (veh/km/lane). The sum of the lane traffic densities represents the total traffic density on the highway. The traffic density is very commonly used as a traffic measure because it gives a better indication of traffic flow quality than traffic flow and mean speed [133].

There are some others traffic variables which are less important than the previous three variables, such as occupancy, time headway, and distance headway. The occupancy of a highway or a lane is quantified as the part of the time in which the detector detects a vehicle. The time headway (h) between two vehicles is measured as the time between the arrival of the front of a vehicle and the arrival of the front of the next vehicle at a certain point, while the distance headway (km) between two vehicles is measured as the distance between the front of a vehicle and the front of the next vehicle. That means the length of the first vehicle is included in the distance.

2.2.2 Sensor Systems

The traffic variables described in the previous section are measured using a wide variety of technologies [92] such as pneumatic detection systems, inductive loops, cameras, ultrasonic sensors, microwave sensors, active and passive infrared sensors, passive acoustic arrays, magnetometers, etc. In this section, the three most widely used technologies (i.e. inductive loops, traffic cameras and pneumatic tube detection systems) are briefly discussed. A more detailed description of traffic sensor technologies and a comparison of their performance can be found in [92] and [93].

Inductive Loops

The inductive loop, which was introduced around 1960, is the most widespread traffic detection system [115]. It is simply a coil of wire that is put in or on the highway's surface (See Figure 2.1). This wire loop is connected to an electrical circuit which produces an oscillation frequency through the loop. When a vehicle passes over the wire loop (or is stopped within the area enclosed by the loop), the metal parts of the vehicle shifts the oscillation frequency of the electrical circuit. This frequency shift is registered and interpreted as vehicle detection by the controller. The main disadvantage of this type of traffic detection systems is the poor sensitivity of the loop during highway maintenance works.

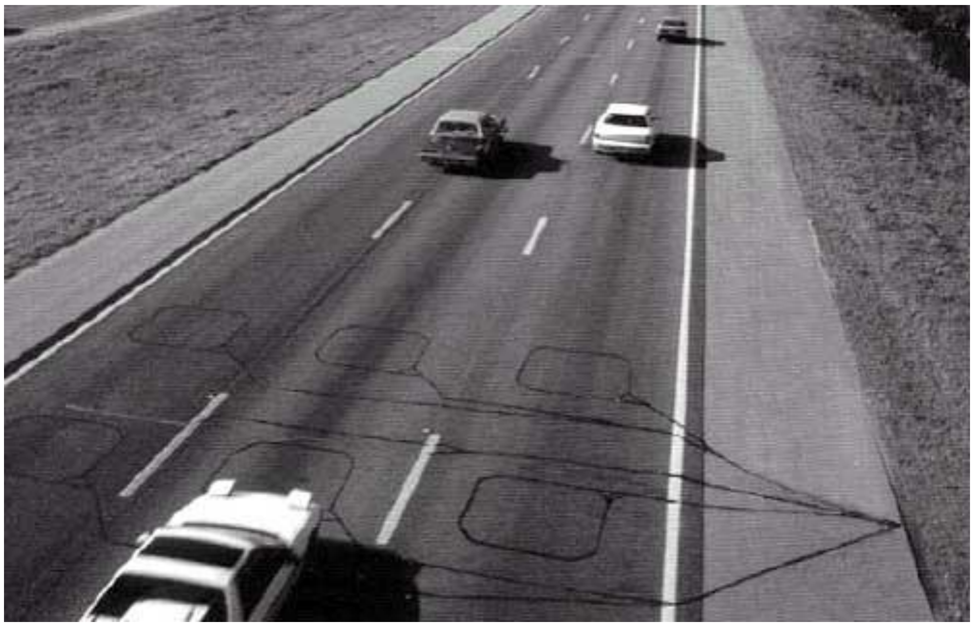


Figure 2.1: Inductive Loop Detector System**[**92**]**.

Traffic Cameras

Traffic cameras were first used as a traffic detection system around 1980 then became increasingly popular [115]. A fixed camera is mounted above the highway to transmit images to a video processing unit that uses image recognition algorithms to extract the desired parameters. Figure 2.2 shows an example of the image a traffic camera produces. The video images sent by traffic cameras can also be used by the traffic centre as a visual inspection tool. The main advantage of using traffic camera as a traffic detection system beside the visual inspection possibilities, is the higher average accuracy of the measurements. However, the high cost of its installation and the weakened accuracy during low visibility weather conditions are the main disadvantage of a traffic camera detection system [93].

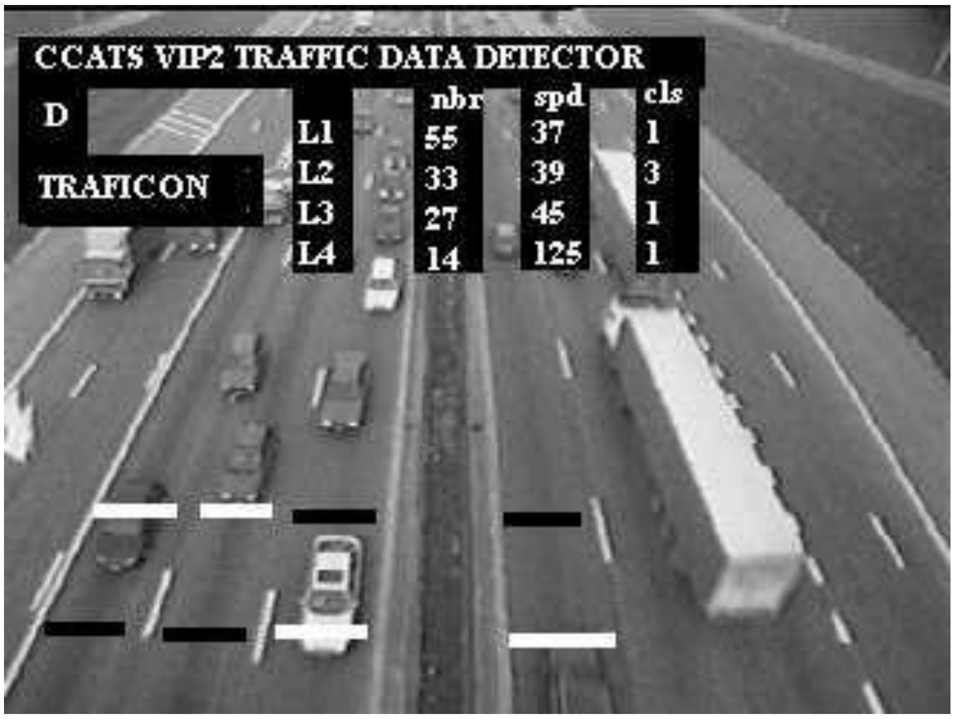


Figure 2.2: An example of a video image of a highway taken by a traffic detector camera **[**9**]**.

Pneumatic Traffic Detection Systems

Pneumatic traffic detection systems [88] consist of a tube that is installed on the surface of the highway as shown in Figure 2.3. When a vehicle passes over the tube, it makes pneumatic shock waves in the tube which are detected and processed. The detected pneumatic shock waves and the delays between sequential shocks are used to predict the traffic flow and its speed. In addition, it can be used to classify the vehicles. The ease of installing pneumatic tube detectors makes the mobility of the installation the main advantage of this traffic detection technology. One disadvantage however, is their limited accuracy. Moreover, the pneumatic sensors are not suitable for long term operation because the pneumatic tubes are laid on the highway's surface which makes them susceptible to wear and tear due to frequent use.

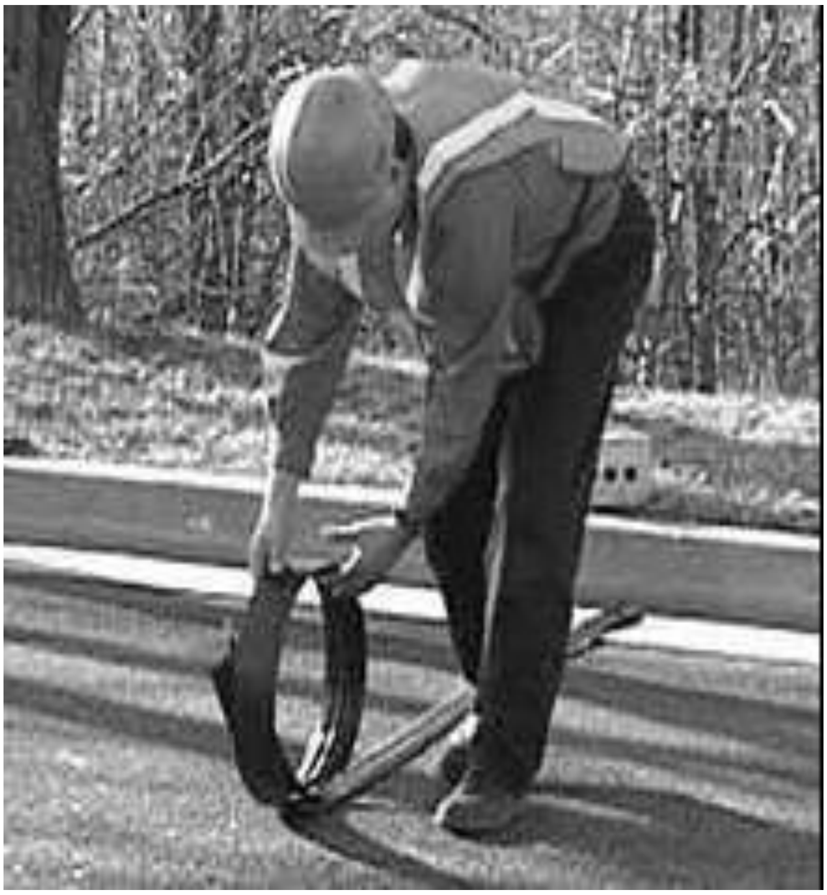


Figure 2.3: The installation of a mobile pneumatic tube traffic detector. The tube is fixed on the highway surface and connected to a registration unit at the roadside [9].

2.2.3 Traffic Control Actions

Traffic control actions (or measures) are the strategies that are applied, or could be applied, in order to improve traffic performance. In this section we give a brief description of some control actions. The control actions listed below are not all the possible traffic control actions, but they are the most widely used.

Ramp Metering:

Ramp metering uses a traffic device, usually a basic traffic light (red, amber, green) or a two-phase light (red and green) to control the flow of traffic entering highways according to current traffic conditions by determining the flow rate at which vehicles can enter the highway (See Figure 2.4). It is one of the most frequently investigated and applied highway traffic control actions. Usually, the main goal of the ramp metering system is to avoid congestion and to reduce the vehicle's total travel time. A traffic light is used to control the flow at the on-ramp by determined the flow rate using appropriate red, green and amber light timings. For example, sensors detect how many vehicles are travelling on the highway. When they detect that the traffic is reaching capacity, the ramp metering system is turned on. The implementation and effects of the ramp metering control action are studied in detail in [126] and [82].

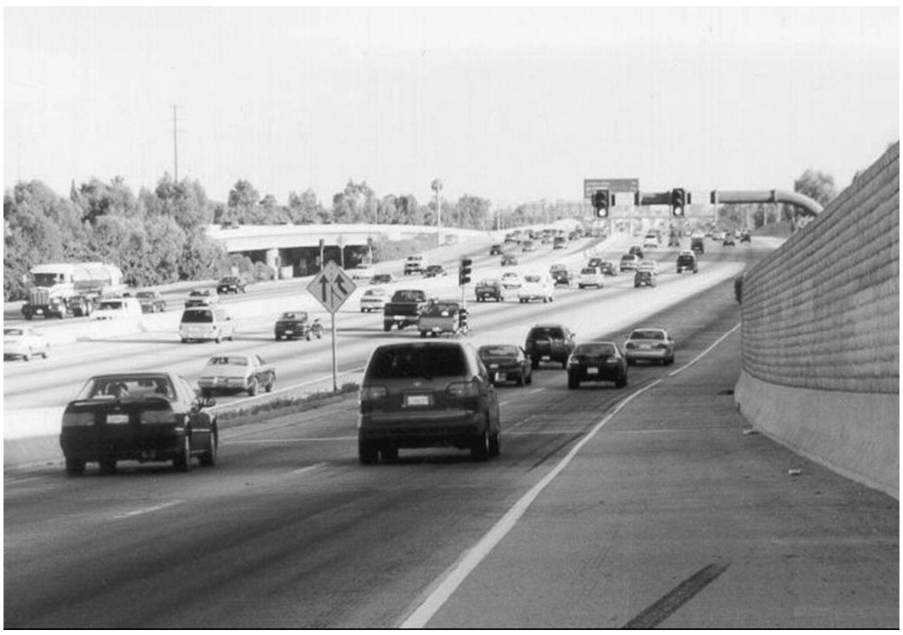


Figure 2.4: An example of ramp metering system (freeway-to-freeway metering) [92].

Variable Speed Limits

Most advanced highways are provided with variable speed limits (VSL) (See

Figure 2.5). A road speed limit means the maximum allowed speed for road vehicles. VSL is used as a control action to increase safety by reducing the speed limits upstream of congested areas [174] and [143].

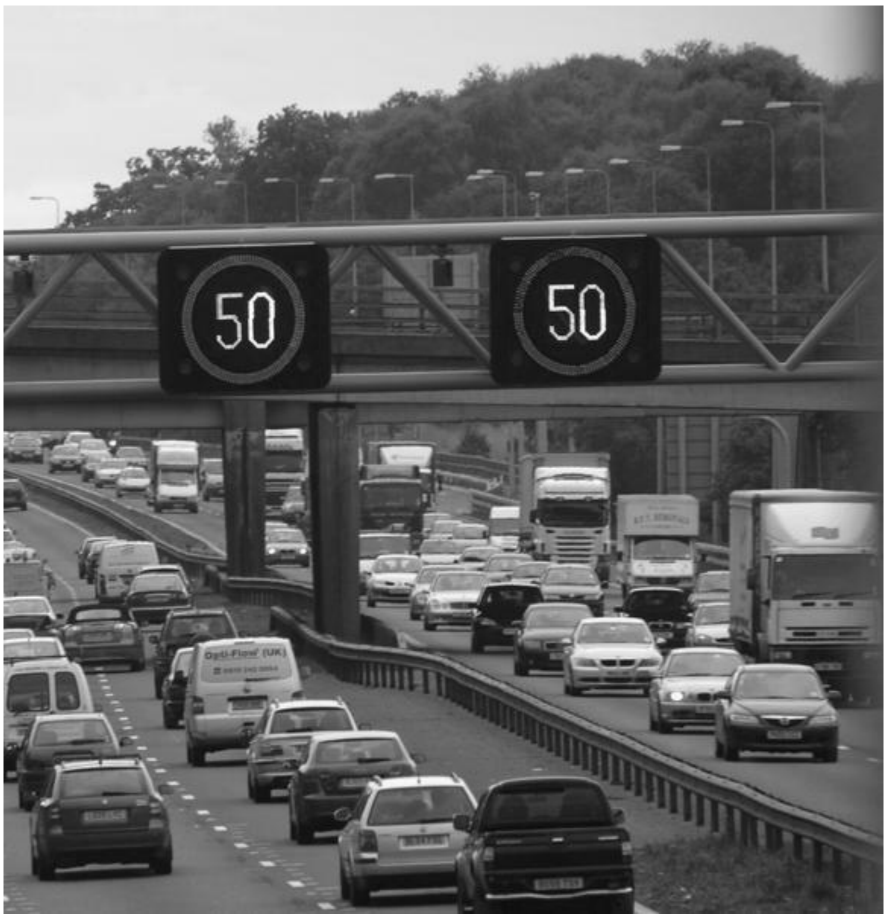


Figure 2.5: An example of variable speed limits sign setup on a highway **[**185**]**.

Route Guidance

When more than one alternative route is available to the same destination, route guidance systems are applied to help (or direct) travellers to choose their optimal route. Route guidance systems use Variable Messages Signs (VMSs) to display traffic information such as vehicles queue length and delay on alternative routes, or to ask vehicles to take alternative routes [125]. (See Figure 2.6).

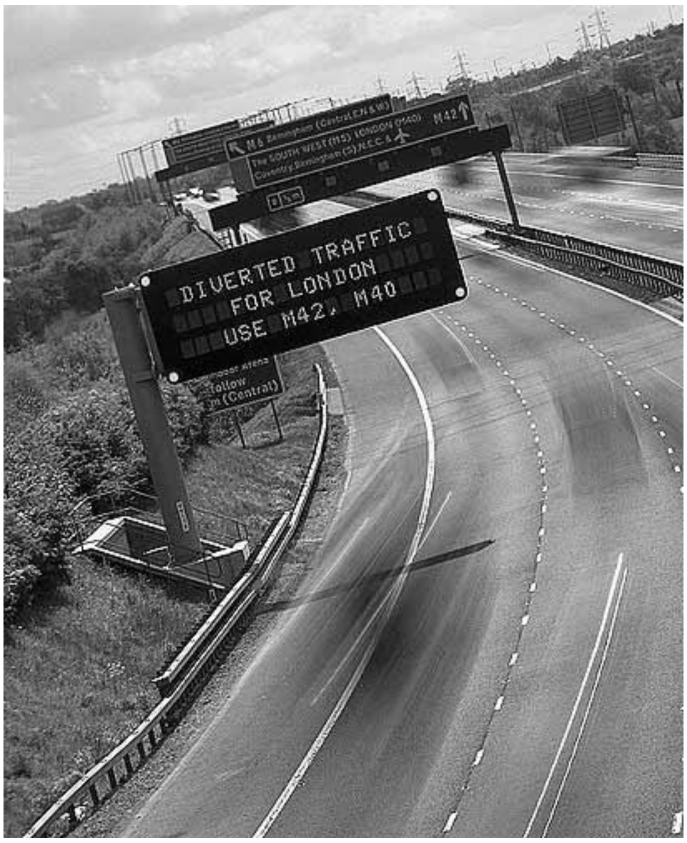


Figure 2.6: An example of variable messages signs setup on a highway **[**9**]**.

Hard Shoulder Opening (or Peak Lanes)

The hard shoulder lane of a highway, which is usually used by emergency vehicles, can be opened for all vehicles as an additional lane during rush hours. VMSs show a green arrow when the lane is opened or a red cross when it is closed (See Figure 2.7). Sometimes, the hard shoulder lane is opened for only dedicated vehicles, such as freight transport and public transport in order to decrease the impediment that congestion causes to these vehicles. Hard shoulder opening is useful as a control action when the additional capacity is needed on a specific section of the highway to reduce congestion and when the downstream highway infrastructure can accommodate this. However, using the emergency lane as a normal lane reduces the safety level [114].

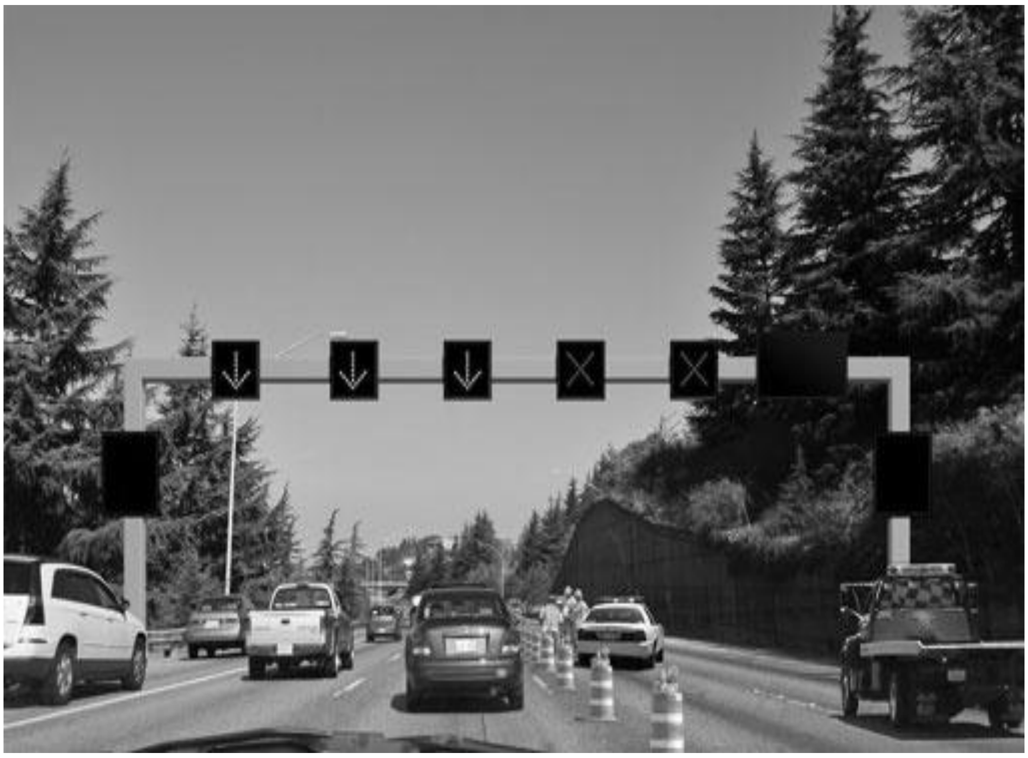


Figure 2.7: Green arrows and red crosses on VSL show which lanes are open and which are closed **[**178**]**.

Bi-Directional Lanes

A bi-directional lane is a highway lane that can be used in both directions. The direction is determined using VMS (showing a red cross or a green arrow) based on the direction of the highest traffic demand. This traffic control action is applicable when the traffic demand in the highway is high in just one direction [114].

The "Keep Your Lane" Directive

When a reduction of disturbances in the highway traffic flow is required in order to prevent congestion, the "keep your lane" directive is displayed on VMSs to direct the drivers not to change lanes. This control action is appropriate when the traffic flow is most unstable [114].

2.3 Traffic Flow Modelling

The modelling of traffic flows on highway networks is a useful tool for several traffic management tasks, including:

- The development and evaluation of traffic control actions.
- The prediction and observation of traffic conditions in complex traffic networks.
- The assessment of the impact of new transportation facilities, comparison of alternatives, etc.
- The assessment of the impact of capacity reducing events (e.g. road maintenance) or increased demand, etc.
- The training of new operators.

Traffic flow can be described as consisting of many moving particles that interact with one another, as well as with the environment. Traffic flow modelling can be performed using either microscopic or macroscopic parameters. Since we will use traffic modelling later on in this study, this section briefly describes the fundamental characteristics of both microscopic and macroscopic traffic flow models. A detailed overview of traffic flow models can be found in [67], [68] and [191].

Microscopic

Microscopic traffic flow models describe the behaviour of vehicles individually, which allows their users to assign different characteristics to each vehicle. The characteristics of a vehicle include type (e.g. truck, car), destination, and chosen route. Also, the driving style of the driver (e.g. patient, aggressive) can be assigned as a vehicle's characteristics. A microscopic traffic flow model is useful for moderate-sized systems, where the number of vehicles is not very large, and the analysis of the individual behaviour of vehicle is required. Examples of microscopic simulation packages are AIMSUN2 [6], Vissim [39], and Paramics [132].

Macroscopic

Macroscopic traffic flow models describe the traffic flow as a whole, without distinguishing between individual vehicle behaviours such as vehicle type or chosen route. The aggregate terms such as average flow, average density, and average speed are used in macroscopic traffic flow models for describing the traffic. A macroscopic model may be selected for a large scale system with higher density, and when the analysis of the global behaviour of groups of vehicles is required. METANET [113] and MASTER [60] are examples of macroscopic simulation packages. MASTER is a macroscopic model which is based on a gas-kinetic traffic equation [58]. Its equations were derived from a "microscopic" description of driver–vehicle behaviour [59]. The main characteristic of MASTER is its non-local interaction term that takes into account the space requirements of vehicles and the correlations of successive vehicle velocities.

Since, in this study, modelling traffic flow as a whole without any concern for the individual behaviour of vehicles is sufficient to generate the required experimental data, a macroscopic traffic flow model (i.e. METANET) has been selected and used in the experiments described in Chapters 4, 6, and 7. The main reasons for the selection of METANET are its simplicity and the good trade-off between simulation speed and accuracy [122]. The following section briefly discusses the main characteristics of the METANET traffic simulation model.

2.4 METANET

The METANET model, which was developed by Messmer and Papageorgiou [113], is a program for a highway network simulation based on a macroscopic modelling approach. The macroscopic modelling approach allows for the simulation of all kinds of traffic conditions (free, dense, and congested) and of capacity-reducing events (incidents) with prescribed characteristics (location, intensity, and duration).

METANET may be applied to existing or hypothetical, multi-origin, multi-destination, multi-route highway networks with arbitrary topology and geometric characteristics, including bifurcations, junctions, on-ramps and off-ramps. By use of a special modelling option (store-and-forward links), METANET also provides the possibility of considering non-highway links in a simplified way. METANET considers the application of traffic control actions, such as collective and/or individual route guidance, as well as ramp metering and highway-to-highway control, at arbitrary network locations. Several options are offered for describing or prescribing the average route choice behaviour of drivers groups with particular destinations. Route guidance and dynamic traffic assignment considerations in METANET are based on the notion of splitting rates at bifurcation nodes rather than on path assignment. Among other advantages, this approach enables the consideration of route guidance or traffic assignment for a part of the network (rather than the whole network) if so desired by the user.

A highway network is represented by a directed graph with the links corresponding to highway stretches. The simulation of traffic behaviour in the highway links is based on a macroscopic modelling approach with traffic variable density (veh/km/lane), mean speed (km/h), and traffic volume (or flow) (veh/h). Each highway stretch has uniform characteristics, e.g. no on-ramps or off-ramps and no major changes in geometry. Where major changes occur in the characteristics of the highway stretch or in the road geometry (on/off-ramps), a node is defined. Each link is divided into several segments as presented in Figure 2.8.

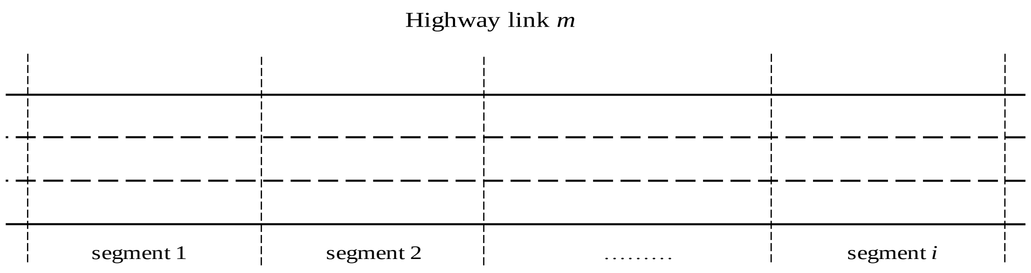


Figure 2.8: A section of a highway modelled by METANET.

Essentially the simulation is fed with demands at its boundaries (inflows) plus origin-destination information (if necessary). These data, together with other values, act as the network boundaries (speeds at main inflows and traffic densities at main outflows), and are called boundary data (or input traffic data in the following network). The origin of the data may be from measurements (if real traffic situations are reconstructed), or the data may be hypothetical (if certain types of traffic situations are studied). Each origin-destination couple may be connected by one or more routes. Based on the network topology, METANET automatically finds all possible loop-free routes. The route choice behaviour inside the network is described by the use of splitting rates which express the portion of drivers deciding at a bifurcation node to use a certain alternative output link towards their destination. Splitting rates can be looked upon as turning rates (the ratio of the traffic volume in each output link of a node) by destination.

METANET (off-line) offers the option to model incidents and a number of different types of control actions such as Variable Message Signs (VMS), Lane Closures, Shoulder Lane Opening, Variable Speed Limitations, Ramp Metering and Traffic Lights at on- and off-ramps.

Simulation results are provided in terms of macroscopic traffic variables such as traffic density, traffic volume, and mean speed at all network locations, as well as in terms of travel times on selected routes. This is done on a configurable output time interval that is usually chosen to be significantly longer (typically several minutes) than the simulation time step (typically 5 to 20s). The display of results is provided both by the time trajectories of selected variables and by graphical representation of the whole network. Global evaluation indexes such as total travel time, total distance travelled, total fuel consumption, total waiting time at network origins, total vehicles driven out network, etc. are also calculated.

2.5 Summary

In this chapter a general discussion and background of the main related topics of road traffic management has been given. We also presented the main features of the traffic simulation model METANET that will be used in the experiments described in Chapters 4, 6, and 7. It is important to note that the choice of the specific traffic simulation model (METANET) presented in this chapter was not imposed by our proposed Intelligent Traffic Control Decision Support System (ITC-DSS). So, any traffic simulation models like METANET could also have been used in this study.

The next chapter will present a survey of the existing work in the field of intelligent road traffic management.

Chapter 3

Literature Review

3.1 Chapter Overview

In recent years, there have been many attempts made to improve road traffic management and control systems as this is one of the major transportation management problems. Artificial Intelligence (AI) based methods are very popular among road traffic management and control applications. This chapter presents a review of the literature related to intelligent road traffic management and control.

We start with a brief background of related intelligent techniques including fuzzy logic, neural networks, genetic algorithms and intelligent multi-agents. Then an overview of the research that has been done so far with regard to intelligent road traffic management systems and control areas including traffic simulation and prediction systems, traffic problem detection systems and traffic control systems will be provided in Section 3.3.

3.2 Background to AI Techniques

This section gives a brief description of some AI techniques used in the road traffic management and control field, including fuzzy logic, artificial neural networks, genetic algorithms, and intelligent multi-agents. The AI techniques described in the following section are not all the AI techniques used in road traffic management and control, but they are the most popular and have been employed in this study.

3.2.1 Fuzzy Logic

Fuzzy logic, which was introduced in the mid-1960s by Dr. Lotfi Zadeh [188], is a problem-solving control system methodology which allows us to measure imprecise and dynamic factors in order to achieve a reasonable judgment based upon vague, ambiguous, imprecise, noisy, or missing input information. Fuzzy Logic considers more than binary, either-or choices. It analyses analog input values in terms of logical variables over the range 0.0-1.0, with an infinite set of values. For example, when we say that it is hot at 32◦C. Does that mean it is not "hot" at 31.99◦C? There is a range over which we can say it is hot. Fuzzy logic recognizes that range and takes into account any related factors such as humidity, in order to generate an optimal temperature output. Although fuzzy logic theory was introduced in the mid-1960s, the theory was not applied commercially until 1987 when the Matsushita Industrial Electric Co. in Japan used it to control water temperature in a shower head [41]. Now, fuzzy logic is widely used in machine control applications. It is used to automatically optimize the wash cycle of a washing machine by sensing the load size, fabric mix, and quantity of detergent and has applications in the control of passenger elevators, household appliances, cameras, automobile subsystems, and smart weapons [41].

Fuzzy inference is the process of using fuzzy logic techniques to create an output value from a given input. A fuzzy inference system (also referred to as fuzzy if-then rules) consists of three stages. The first stage, which is called fuzzification, transforms crisp inputs (numerical values) into membership degrees to the different fuzzy sets of the partition. The second stage is the application of fuzzy if-then rules. The system invokes each appropriate rule and generates a result for each, then combines the results of the rules. Finally, the third stage, which is called defuzzification, converts the combined result back into the crisp or actual results. Structure identification and parameter adjustment are the two main tasks involved in building a fuzzy system. The structure identification task creates the initial structure of the fuzzy system, including the determination of input–output space partitions, antecedent and consequent variables of IF–THEN rules, the number of such rules, and initial positions of membership functions. The parameter adjustment task identifies a feasible set of parameters under the given structure. The readers are referred to [187], [40], [183], [96] for more information about the fuzzy logic technique.

3.2.2 Artificial Neural Network

An Artificial Neural Network (ANN), which was developed in the 1950s aimed at imitating the biological brain architecture. It was a flexible mathematical paradigm that is capable of identifying complex relationships between input and output data sets. ANN is a parallel distributed system composed of many highly interconnected parts and is organized into different layers of non-linear processing elements, called neurons [56]. When ANN is operating, each neuron receives many input signals. Then, based on an internal weighting system, it produces a single output that is typically sent as an input to another neuron. The input layer receives the input, and the output layer produces the final output. Generally, one or more hidden layers are involved in between the input and the output layers. The structure of ANNs makes it impossible for the user to predict or know the exact output. This makes ANN to be widely regarded as a 'black box'.

ANNs are more useful and efficient, particularly with regard to problems for which the characteristics of the processes are hard to describe using physical equations. Many ANN-based models have been developed successfully for very different environmental purposes, such as regression analysis, time series predictions, pattern recognition, sequential decision making, clustering, filtering, etc. [36], [73] [35], [85].

An important characteristic of artificial neural networks is their ability to learn without a need to be reprogrammed. ANNs learn by example, just as humans do. An ANN is firstly created with randomized weights for all neurons, which means that the ANN must learn to solve the particular problem for which it is intended. Generally, there are three major methods for learning: 1) Supervised learning (Associative learning) [95] in which an ANN is trained by providing it with input and matching output pairs. These input-output pairs are provided by an external supervisor, or by the system which contains the ANN (self-supervised). A very common and well-known example of this type is the back-propagation ANN; 2) Unsupervised learning (or Self-organization) [7] in which an (output) unit learns to respond to clusters of pattern within the input. In this type of learning, the system should discover statistically salient features of the input population. Different to supervised learning, there is no a priori set of categories into which the patterns are to be classified; rather the system must develop its own representation of the input stimuli. Unsupervised learning is exposed to large amounts of data and tends to discover patterns and relationships within that data. Researchers often use this type to analyze experimental data; 3) Reinforcement learning [156], in which the input data is usually not given but the learning machine performs some actions on the environment and receives a feedback response from the environment. The learning system parameters are adjusted during the learning process, based on the environmental response until an equilibrium state occurs.

3.2.3 Genetic Algorithms

A genetic algorithm (GA), which was first introduced by John Holland in the early 1970s [65], is a parallel and global computing search technique that is inspired by the natural evolution process [44]. Due to the fact that GA simultaneously evaluates many points in the search space, it is more likely to converge toward the global solution. GA is a particular class of evolutionary algorithms which applies operators, inspired by the mechanics of natural selection, to a population of the parameter space at each generation; it explores different areas of the parameter space, and then directs the search to regions where there is a high probability of finding improved performance.

Over the last decade, GAs have been widely used as search and optimization tools in various problem domains, including science, commerce, and engineering. As a general purpose optimization tool, GAs are moving out of academia and finding significant applications in many other venues. Their popularity can be attributed to their freedom from dependence on functional derivatives and their incorporation of these characteristics [31], [166]:

- GAs are parallel search procedures and can be implemented on parallel processing machines.
- GAs are applicable to any optimization problem.
- GAs are stochastic and are less likely to get trapped in local minima as opposed to, for example, gradient descent techniques.
- GAs are flexible for both structure and parameter identification.

The GAs evolution can be summarized as follows: create a population of individuals, evaluate their fitness, generate a new population by applying genetic operators, and repeat this process a number of times [118]. To start the optimization process, GAs generate randomly, or by other means, a population of, say, N individuals. Generally, each individual in the population consists of encoded strings representing a solution. Each solution has a fitness value evaluated by the same objective function and constraint satisfaction. The individuals with higher fitness values are usually selected and sent to a mating pool. Different selection methods, such as roulette wheel selection and stochastic universal sampling, can be used for this operation. Solutions having higher fitness values are most likely to survive for the next generation. A crossover operator is used on these strings to obtain the new solutions that inherit the good and bad properties of their parent solutions. The crossover operator works on randomly selected pairs of selected solutions from the mating pool, with a certain crossover rate. The crossover rate is defined as the probability of applying crossover to a pair of selected solutions. There are many ways of defining this operator such as single point, double point, multipoint and uniform crossover. These traditional crossover operators are discussed in [116], [25]. More information about the GA technique and its applications can be found in [44], [51], [20], [84].

3.2.4 Intelligent Multi-Agent

Agent technology is a fast growing area of research in AI. The agent is defined as “a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives”[176]. Two main attributes should be available in an agent. The first one is being autonomous. This means that an agent makes its own decision in order to achieve its goals as an independent unit. The second one is being situated in a challenging environment with dynamic, unpredictable and unreliable characteristics [190]. An intelligent agent should have, besides AI, the following additional capabilities [177]:

Reactivity: Intelligent agents are able to perceive their environment, and respond in a timely fashion to changes that occur in it in order to satisfy their design objectives.

Pro-activeness: Intelligent agents are able to exhibit goal-directed behaviour by taking the initiative in order to satisfy their design objectives.

Social ability: Intelligent agents are capable of interacting with other agents (and possibly humans) in order to satisfy their design objectives.

When a group of agents are put to use in the same environment to solve a variety of problems, they are usually referred to as multi-agents. Multi-agent Systems (MAS) is a subfield of AI (Artificial Intelligence) that aims to provide both principles for the construction of complex systems involving multiple agents, and mechanisms for the coordination of independent agents' behaviours [151]. Thus, in multi-agent systems, agents are forced to coordinate their activities so as to avoid negative interactions with their acquaintances, and to exploit synergic potentials. The readers are referred to [176], [177], and [94] for more information about the multi-agent technique and its applications in traffic and transportation.

3.3 A Survey of Intelligent Traffic Management Systems

Here we review the different intelligent systems used for road traffic management. In order to present a consistent review of the literature related to intelligent traffic management, we have classified the existing intelligent road traffic management applications into three groups: (1) traffic problem detection systems; (2) traffic simulation and prediction systems; (3) traffic control systems. Based on this classification, the following sections present a brief survey of intelligent traffic management systems.

3.3.1 Traffic Problem Detection Systems

Traffic problems such as congestion, accidents, broken down vehicles, spilled loads, construction activities and temporary maintenance and other unusual events are defined as being events that reduce the capacity of network links to carry traffic.

Early and accurate detection of traffic problems is necessary for the restoration of a smooth traffic flow [149], [71], [159]. Therefore, recently there has been a considerable research effort on the development of intelligent applications to assist the traffic control centres in the task of detecting traffic problems. Various AI techniques have been used for developing traffic problem detection applications. ANN is among the most promising technique. The application of ANNs for incident detection was first investigated in [23] when ANN was used for the automated detection of non-recurring congestion on urban freeways.

Khan et al. [89] have used ANN to develop a classifier system to assist in the detection of different types of traffic operational problems based on detector data from system loops collected on a cyclic basis. The proposed classifier has been developed to be able to detect three types of operational traffic problems including lane-blocking incidents, special event conditions (e.g. sports events, conventions, conferences, or activities that attract a large number of people) and detector malfunctions. The inputs of the classifier model were traffic volume and occupancy over a cycle and over several previous cycles, the difference in occupancy between adjacent lanes, and the differences in upstream link and downstream link occupancy.

Traffic incidents are a major cause of traffic congestion, so incident detection has become an important issue in freeway traffic management systems. Using ANN, Yu et al. [186] have developed a traffic incident detection model based on a fusion of loop detector and probe vehicle data. In the incident detection model they developed, the cumulative sum (CUSUM) approach [53] was used to develop incident detection algorithms using loop detector data and probe vehicle data respectively, while the Back-Propagation neural network was employed to combine the outputs from both incident detection algorithms.

Wen et al. [173] considered incident detection as a kind of pattern recognition problem when they developed their probabilistic neural network (PNN)[146] based on an incident detection model. The PNN based model they developed has been trained and evaluated using different patterns under a variety of flow conditions and traffic periods generated by a traffic simulation model. Another use of PNN for the incident detection problem has been investigated in [80]. The main limitation of using PNN for solving incident detection problems is that a large memory and computation time is required due to the large size of the PNN pattern layer [148].

Srinivasan et al. [149] have evaluated the performance and the adaptability of three promising NN-based incident detection models: a multilayer feed-forward NN (MLFNN), a basic probabilistic NN (BPNN) [146] and a constructive probabilistic NN (CPNN) [11]. These three models were developed on an original freeway site in Singapore and then adapted to a new freeway site in California. The result showed that the MLFNN model had the best incident detection performance at the development site, while the best performance after model adaptation at the new site was recorded for CPNN. Also, the results showed that the adaptation method for the CPNN model is less laborious. Overall, the result of the evaluation showed that the CPNN model is the most portable model for freeway incident detection.

Fuzzy logic has also been used in the development of incident detection applications[100], [171], [147]. For example, Lee et al. [100] have proposed a fuzzy logic based incident detection algorithm for signalized urban diamond interchanges. The fuzzy logic has been employed to detect the incident by evaluating the past five-minute average of the traffic measures (including lane-by-lane volumes, queue length, speed, and occupancy) and the percentage change from the past five-minute average during the most recent minute. The outcome was an incident report including the time, location, and severity of the incident.

In summary, AI based techniques, such as neural networks and fuzzy logic, have shown great potential in the development of automated incident detection algorithm with the promise to give high incident detection rates. However, the existing AI based incidents detection algorithms still have many drawbacks. First, in the neural network based algorithm such as [23], [89], [186] and [173], it is difficult to understand the meanings of neural-network operations, since it is a black-box approach. Also, the implementation of neural networks requires large traffic historical data sets and the state range covered by those data must be wide and large enough. Second, the fuzzy logic based algorithm (such as [100], [171], [147]), as its name implies, does not give a clear “incident” or no incident” signal, rather it gives the likelihood for an incident. The fuzzy logic also requires extensive calibrating to define the logic boundaries.

In general, despite the considerable research effort on the development of automated incident detection algorithms, several studies show that in many traffic management centres, the automatic incident detections have been disabled, because their operational performance has generally been poor in large-scale deployments [128]. For example, a very recent survey [175], which has been designed to evaluate the current status of the incident detection algorithms in the U.S., shows that 90% of survey respondents (i.e. professionals who are involved in freeway and congestion management) felt that the current methods of incident detection are still insufficient.

3.3.2 Traffic Simulation and Prediction Systems

The prediction of traffic condition (e.g. traffic flows, traffic speed, travel times, etc.) is currently one of the fundamental tasks of traffic management and control [119]. The results of traffic condition predictions can be used for different purposes such as for managing traffic congestion, influencing travel behaviour, and generally improving the performance of the traffic management system. Most of the existing traffic simulation and prediction applications are statistically-based models, which use a mathematical formula for predicting the traffic state such as are shown in [142], [167], [164], [43]. However, these statistically-based models can only fit traffic flow well under ideal physical environments but may not work satisfactorily in certain complex situation because of their strict mathematical assumptions [131]. In consequence, a considerable research effort has been done recently that has focused on the use of AI techniques in the development of traffic prediction applications, e.g. [131], [32], [79], [184].

ANNs have been widely used for predicting and simulating traffic flow. Qiao et al. [131] have proposed an auto-adaptive model for simulating and forecasting the dispersion of traffic flow on road segments using neural network-based system identification approaches. Their proposed neural network-based model can adapt itself to a wide variety of traffic situations by adjusting the structure and linking the weights of the neural network-based model on-line. Data simulation and field-testing showed a reliable performance of the proposed neural network model compared with classic mathematical methods.

Dia [32] has proposed an object-oriented neural network model for short-term traffic condition forecasting. The proposed object-oriented neural network model was developed using speed and flow data collected from four inductive loop detector stations installed on a 1.5-km section of the Pacific Highway in Queensland in order to predict speed up to 15 minutes into the future. The results showed a high degree of accuracy (90 – 94 %) when predicting speed data up to 5 minutes into the future. Then the accuracy of the models was decreased to 88% and 84% for prediction horizons of 10 and 15 minutes, respectively.

Jiang et al. [79] have also used ANN to develop a traffic prediction model based on the relationship of segment traffic volume and average travel time. The model has been developed for the urban traffic network installed self-adapted traffic control system. The outputs of ANN were traffic volume and average speed. The average speed was then used to calculate the average travel time. The approach was tested using the data from an arterial road in Changchun, China and results showed the reliability of the proposed ANN-based model in terms of speed in computation and economy.

Recently, the Fuzzy Neural Networks (FNN) technique has been applied successfully to the prediction of traffic conditions. Yin et al. [184] have developed a fuzzy-neural model to predict traffic flows in an urban street network. The proposed system consists of two modules: a Gate Network (GN) and an Expert Network (EN). The GN classifies the input data into a number of clusters using fuzzy logic, and the EN is used for specifying the input-output relationship as in a conventional neural network approach. The parameters of the model are adaptively adjustable in response to the real-time traffic conditions. The results showed that the Fuzzy-Neuro Model (FNM) outperforms the conventional Neural Network Model (NNM). FNM also showed better performance in computing time than did NNM. The developed system was tested both by simulation and with real observation data.

Quek et al. [133] have developed a novel approach for modelling traffic behaviour using a specific class of self-organizing fuzzy rule-based system known as the Pseudo Outer-Product Fuzzy-Neural Network using the Truth-Value-Restriction method (POPFNN-TVR). POPFNN-TVR was used to predict the speed and density of a particular lane in a three-lane highway, given the speed and density of the other two lanes. The prediction capability of POPFNN-TVR was validated against a back-propagation NN and the results were promising. Other FNN-based traffic prediction models have been presented in [184], [134], [127] and [49].

Multi-agent system technology has also been adopted in constructing simulation models for road traffic management. For example, Donieca et al. [34] have developed a multi-agent behavioural traffic simulation model based on the opportunistic individual behaviours that describe the norm violations, and the anticipatory individual abilities of simulated drivers that allow critical situations to be detected. The proposed model has been validated for different traffic scenarios by simulating the traffic in a real intersection and then comparing the simulated traffic flow with the real flow. Another multi-agent-based road traffic simulation model has been proposed by Meignan et al..[112]. They have developed a bus-network simulation tool using a multi-agent approach, to analyze and evaluate a bus-network at diverse space and time scales. The proposed tool describes the global system operation as behaviours of numerous autonomous entities such as buses and travellers.

In summary, the use of artificial neural networks in the development of traffic prediction application is very promising and has several advantages over classic mathematical models. Since neural network based models rely on the self-evolved parameters and recurrent calculation under the teacher signal more than the complicated mathematical analysis, it is easier to process and also shows better robustness under the noisy environment (e.g. [79]). Furthermore, neural network based traffic prediction models have a good adaptive ability since the parameters can be adjusted automatically as the OD (original and destination) changes in the environment such as in [131]. However, the lack of traffic information provided by neural network based traffic prediction models comparing with mathematical models (especially microscopic models) is the main disadvantage of neural network based models.

3.3.3 Traffic Control Systems

Traffic control is one of the fast growing areas related to traffic management problems. For example, when traffic problems are detected, an efficient control method is needed to assist traffic control operators in their assessment of the severity of that problem, and also to help them to make the best decisions with regard to solving that problem. Traffic control involves different types of applications such as traffic signal and lights control systems, traffic control decision support systems, intersection management systems, ramp metering control systems, and so on.

The feasibility of using AI-based systems has been proven for traffic control applications for over 15 years, during which some operational knowledge-based expert systems were developed [37],[192]. Based on Ritchie's investigation [140], the Santa Monica Freeway Smart Corridor project [37] was one of the very first times that a real-time knowledge-based expert system was employed for traffic control purposes. In 1994, Zhang and Ritchie [192] proposed a knowledge-based decision support system called FRED (Freeway Real-Time Expert System Demonstration) for assisting traffic operations centres in detecting, verifying, responding to, and monitoring, non-recurring congestion on urban freeways in Orange County, California.

TRYS is another well-known knowledge based traffic control application which has been described in [29] and [28]. TRYS is an agent-based environment for building intelligent traffic management systems applications for urban, interurban and mixed traffic areas. It provides a generic and modular knowledge model supporting an intelligent reasoning layer that can complement conventional traffic management application capabilities. In the TRYS systems, the traffic network is divided into overlapping regions, called problem areas, and each problem area is supervised by an agent. Each agent has to detect and diagnose its traffic problems and subsequently propose possible control measures to a higher level agent, whose aim is to produce global proposals for the whole traffic network. InTRYS [27] and TRYS2 [124] are examples of TRYS-based applications.

In the later stages, with the arrival of advanced technology, fast computers and the significant research effort in the traffic control domain, several AI approaches have been used for traffic control applications. Hegyi et al. [57] have presented a fuzzy logic-based decision support system (FDSS) for helping operators to manage non-recurrent congestion. Their proposed FDSS was part of a larger traffic support system, which uses a case-base and fuzzy logic to suggest to the operators of the traffic control system whether or not a particular control measure should be activated. In FDSS, the case-base is constructed off-line using a traffic simulated model and fuzzy logic is employed in retrieving cases from the case-base. Since the proposed FDSS does not use any knowledge from experts, or heuristic rules, and it is only based on a case-based reasoning system, the quality of its results depends basically on the quality of the case-base. FDSS has been extended by De Schutter et al. [30] using a multi-agent framework, where the network was divided into sub-networks and each sub-network has its own case-base and evaluation agent.

Katwijk et al.[165] have proposed a software environment for the rapid development of multiagent control systems in road traffic management. The proposed tool consists of an interaction model, intelligent models, and a world model. The interaction model was used to model the interactions between the agents. The intelligence models were used to model the intelligence of the agents that collectively give shape to the traffic control process. The world model was used to represent the outside world, i.e. the traffic process.

Developing real-time freeway traffic routing strategies that suggest routes to drivers to optimize the utilization of network capacity, is a very challenging task. Sadek et al. [141] have examined the potential for using case-based reasoning (CBR) to overcome this task. In their study, a prototype CBR routing system for interstate network in Hampton Roads, Virginia, was developed. The case-base of the proposed CBR system, which consists of routing scenarios that have actually been implemented in the real world and their outcome assessed, was created. The results showed that the prototype system is capable of running in real-time, and of producing high quality solutions using case-bases of reasonable size.

Traffic signal control, which is usually implemented to reduce (or eliminate) traffic conflicts at intersections, is a control problem with a number of complex variables and objects. Intelligent strategies for traffic signal controls that detect and respond to traffic in real time, can provide significant decreases in terms of flexibility, adaptability and possibility with regard to handling uncertain information and dealing with conflicting situations [109]. Consequently, several AI-based approaches have been developed for traffic signal control applications. Fuzzy logic is among the most popular techniques.

Trabia et al. [161] have developed a fuzzy logic-based adaptive traffic signal controller for an isolated four-approach intersection with through and left-turning movements. Using vehicle loop detectors placed upstream of the intersection on each approach, the fuzzy controller measures approach flows and estimates queues at regular time intervals. The information with regard to these measurements is used in two-stage fuzzy logic procedures to decide, at any given time, whether to extend or terminate the current signal phase. The traffic intensity on each approach, which is estimated by the controller in the first stage, is used in the second stage to determine whether to extend or terminate the current phase. The performance of the two-stage fuzzy controller was compared to that of the traffic-actuated controller for different traffic conditions on a simulated four-approach intersection. The results showed that the two-stage fuzzy logic controller in particular produced a better performance under non-recurring traffic conditions.

Beauchamp-Baez et al. [8] have proposed an intelligent approach for traffic signal control. The proposed approach is a fuzzy logic based phase sequencer (PS) for signalized intersection control. The phase sequencer operates in conjunction with the Fuzzy Logic Controller for Traffic Systems (FLC-TS). The FLC-TS was employed to determine the desirability of change of phase, based on traffic demand and the time elapsed since the last time manoeuvre was attempted., The PS then decides when to finish a phase and also selects the next phase among the various possibilities. The results of the comparison did not illustrate a significant difference between the PS + FLC-TS and the FLC-TS. The adaptive tuning of the rule base and the membership functions of the PS might result in a better performance.

Wei et al. [172] have presented a fuzzy logic adaptive traffic signal controller for an isolated four-approaches intersection with through and left-turning movements. The controller has the ability to make adjustments to signal timing in response to observed changes. Also the “urgency degree” term, which can describe the different user’s demands for green time, is involved in the fuzzy decision making algorithm. The three level model of fuzzy control was used to determine whether to extend or terminate the current signal phase and select the sequences of the phases. Results showed a better performance for the fuzzy controller than for the traffic-actuated controller.

Fuzzy Neural Networks (FNN) have also been employed for developing a number of traffic signal controllers [83], [189], [106]. For example, Kaedi et al. [83] have proposed a neuro-fuzzy based two-stage method for intersection signal timing control. The first stage uses a neuro-fuzzy network called Adaptive Neuro-Fuzzy Inference System (ANFIS) to predict traffic volume, while the second stage uses a combination of self-organizing and Hopfield neural networks to estimate appropriate signal cycles and optimized timing of each phase of the signal.

The formulation of the fuzzy rules is the most popular problem facing the development of fuzzy logic-based traffic signal controllers. Evolutionary algorithms seem to have achieved some degree of success as a fuzzy rule generator of different kinds of fuzzy signal controls. Hu et al. [70] have employed a genetic algorithm to generate the fuzzy rules base for their proposed fuzzy controller, using real statistical traffic data for the intersection. They have proposed a fuzzy logic traffic controller to adjust the time parameters and phases of traffic signals at a single real intersection, which consists of multiple lanes with turns. The lanes are organized into a number of groups controlled by individual traffic lights. These lights are further arranged into several light phases. A fuzzy controller was developed to control the length of time of each light phase. Other uses of GA for generating fuzzy rules for road junction traffic control systems are presented in [21] and [63].

Ramp metering or ramp controlling, which is a technique to limit the number of vehicles entering a freeway, represents another main issue in terms of traffic control. Usually, the main goal of the ramp metering system is to avoid congestion and reduce a vehicle's total travel time. Several authors have proposed different intelligent models for ramp control [15], [14], [193]. For example, Bogenberger et al. [15] have applied a self-adaptive fuzzy logic system for ramp controlling problems. The adaptive fuzzy logic algorithm has been used to determine the traffic responsive metering rate. The fuzzy parameters were periodically adapted by an evolutionary algorithm every 15 minutes to guarantee the quick on-line calibration of the existing parameters in a new environment. The developed system was tested with the simulated model on 25 km of the A9 Autobahn (highway) in Munich, Germany. The results of the simulation were satisfying.

Zhang et al. [193] presented another application for ramp control. They extended the Linear Quadratic Regulator (LQR) method [74] by proposing a new traffic-responsive, coordinated ramp control strategy. This control strategy was nonlinear, and realized by a series of neural networks. The parameters of the neural networks were obtained by off-line optimization procedure. The algorithm they have developed showed promising results in reducing total travel time, especially in situation where drastic change in traffic demand and road capacity occur.

In summary, much attention has been paid in recent years to the study and development of intelligent road traffic control systems. The application of intelligent methods for traffic signal control and ramp metering has proven to be worth researching. According to the research reviewed in this subsection, the main research gap in the area of road traffic control is the limitation of the researches that directly investigates the application of intelligent decision support systems for helping human operators to identify the most suitable control action to manage the current traffic state. The identification of a suitable control action for a given non-recurrent traffic congestion situation is a very complex task, which requires expert knowledge, much experience and fast reaction.

There are some researches that offer decision support system for road traffic control [192], [140], [29], [57],[30]. In some of them, (for example, [192], [140], [29]) traffic control decision support systems are part of large knowledge-based traffic management applications. These knowledge-based applications have been developed using only knowledge from experts. This knowledge may vary from person to person, and from time to time. Consequently, the quality of their results depends basically on the quality of the knowledge bases. Moreover, these systems will not be able to generate decisions in cases that are not explicitly covered by such knowledge bases.

On the other hand, traffic decision support systems presented in [57] and [30] do not use any knowledge from experts or heuristic rules and they are only based on a case-based reasoning systems. Thus, the quality of the case-base mainly determines their performance. The disadvantage of these case-based systems is that expert knowledge (if available) is not utilised.

On-line simulation applications are used in some cases for evaluating the performance of traffic control actions. However, simulating different traffic control actions for a number of control measures in a complicated situation is very time-consuming. Using the capability of fuzzy reasoning in handling uncertain information and the capability of neural networks in learning from data, in this thesis we propose an intelligent fuzzy neural network-based traffic control decision support system for road traffic management to assist the human operator to manage the current traffic state in real-time. The main advantages of the proposed system are that it can be developed using numerical data in the absence of expert knowledge, as well as its capability to take advantage of the expert knowledge and heuristic rules when become available.

3.4 Summary

In this chapter we have surveyed a range of the existing works in the field of intelligent road traffic management and control and have discussed common techniques used in these works. The chapter started with a brief background of the main related intelligent techniques, followed by a survey of the related work for traffic simulation and prediction systems, traffic problem detection systems and traffic control systems.

The following concluding remarks can be based on the information presented here: 1) there has been a considerable research effort on the development of incident detection algorithms. However, there are still insufficient applications of incident detection at the present time; 2) ANNs have been widely and successfully used for predicting and simulating traffic flow as an alternative to statistically-based models; 3) most of the most exciting traffic control decision support systems are knowledge-based systems. The quality of the results depends mainly on the quality of linguistic (and other) information provided by experts. 4) the application of intelligent methods for traffic signal control has proven to be worth researching. Of particular interest is fuzzy logic when combined with other techniques, because using FL can help to reduce the required traffic historical data sets

Chapter 4

An Intelligent Traffic Control Decision Support System

4.1 Chapter Overview

In this chapter we introduce an Intelligent Traffic Control Decision Support System (ITC-DSS) which we have developed for road traffic control centres to assist the human operator to manage the current traffic state in real-time. The proposed ITC-DSS receives a large number of possible control actions and uses a pre-trained Fuzzy Neural Network Tool (FNN-Tool) to produce a ranked list of appropriate control actions based on the current traffic state.

Since the proposed ITC-DSS employs a fuzzy neural network, Section 4.2 provides a brief summary of this hybrid intelligent technique. The framework and process of the proposed ITC-DSS is described in Section 4.3, including the structure and the function of the FNN-Tool. The technical feasibility of the proposed ITC-DSS is tested using a case study of a section of the ring-road around Riyadh, Saudi Arabia and the results are presented and discussed in Section 4.4.

4.2 Overview Fuzzy Neural Networks

Fuzzy Neural Network system (FNN) is a hybrid intelligent system which combines the capability of fuzzy reasoning in handling uncertain information and the capability of neural networks in learning from data from the processes. FNN is a fuzzy system that uses the learning ability of the neural networks to determine its parameters (fuzzy sets, fuzzy memberships and fuzzy rules) by processing data. It is suggested by Hayashi and Buckley [54] that any rule-based fuzzy system may be approximated by a neural network, and any neural network (feedforward, multilayered) may be approximated by a rule-based fuzzy system. This type of relation between neural networks and fuzzy rule-based systems is also investigated in [17], [18], [10]. Both fuzzy systems and neural networks are soft computing techniques to model expert behaviour. They are dynamic, parallel processing systems that estimate input-output functions by learning from experience with sample data and without any mathematical models. [117] .

Various fuzzy neural network control schemes have been reported earlier in the literature. These can be classified into two major categories based on the types of fuzzy reasoning and the if-then rules employed; (i) Mamdani type [110], where both the antecedents and consequent parts of the if-then rules are defined in terms of fuzzy sets and membership functions. The knowledge in this type is represented as:

$$R^i : If\ x\ is\ A\ and\ y\ is\ B\ \ then\ z\ is\ C$$

where R^i (i=1,2,….,l) is the ith fuzzy rule, A, B and C are fuzzy memberships associated with linguistic values, x and y are the inputs, and z is the output. Examples of this type are POPFNN [137] and HyFIS [90]; (ii) Takagi-Sugeno type [158], where the consequent part of the if-then rules is a "crisply" defined function, which can be constant or linear with respect to the crisp values of the input variables. This can be viewed as the expansion of a piecewise linear partition represented as:

$$R^i : If\ x\ is\ A\ and\ y\ is\ B\ \ then\ z = a + bx + cy.$$

ANFIS [78] is a very well known example of the Takagi-Sugeno fuzzy neural network type. Mamdani fuzzy inference systems are usually used when the linguistic interpretability of a solution is more important for the user, while Sugeno-type systems are more suitable when the user is concerned about obtaining a more accurate solution and the user is not very worried about its linguistic interpretability. [117]

Fuzzy neural network systems have been successfully applied to a wide range of knowledge engineering and scientific applications such as classification, identification, control systems, decision–making, pattern recognition, and image processes, etc. (see e.g.[42],[108],[69],[111]). A Fuzzy Neural Network has been also employed in the traffic management field as described in several papers. Henry et al. [61] have developed a neuro-fuzzy control method for controlling traffic lights at an intersection. The system offered good results for simple and medium-complex intersections, but poor performance at a complex intersection. Another fuzzy neuron network system has been proposed by Quek et al. in [134] with regard to the analysis and prediction of traffic flow. The system has been fully trained and subsequently used for short-term traffic flow predictions. The prediction results are shown to be promising. Another use of fuzzy neural networks in the field of road traffic management systems are discussed in [184], [13] and[83] (see the previous chapter for more detail).

In this chapter an intelligent decision support system has been developed using a Mamdani type based fuzzy neural network.

4.3 The Proposed Decision Support System

4.3.1 Overall System Framework

As mentioned earlier, a large number of factors which determine the current traffic state need to be considered by the operator as part of the decision making process. These input factors are usually measured by on-line monitoring systems using sensors, detectors and cameras (alternatively, the traffic state can be forecast by a traffic flow simulation model). These input factors include time/day, traffic densities, flows, speeds, inflow demands, outflow restrictions, incidents status, etc. Similarly, there are many possible control actions that can be employed to control the road network, depending upon the nature of traffic problems and the available road control facilities.

The proposed ITC-DSS receives the current values of the input factors (e.g. from a monitoring system) and possible control actions (e.g. from the human operator). ITC-DSS then outputs a ranked list of those control actions based on their performance, to assist the human operator of the traffic control centre to manage the traffic network in real-time. The overall structure of the proposed ITC-DSS is depicted in Figure 4.1. The proposed intelligent system works as follows. Let S be a set of the possible control actions which can be used to control the road network under consideration. S is created for a given road network off-line using the available road control facilities, the traffic operator's experience, and historical data. This also takes into consideration the interrelations between the traffic control actions at different locations in the network. Control action ($ca_i \in S$) can be one control action such as lane closure, ramp metering, shoulder opening, changing speed limits by Variable Speed Limitations (VSL), controlling traffic using Variable Message Signs (VMS), etc, or a combination of control actions.

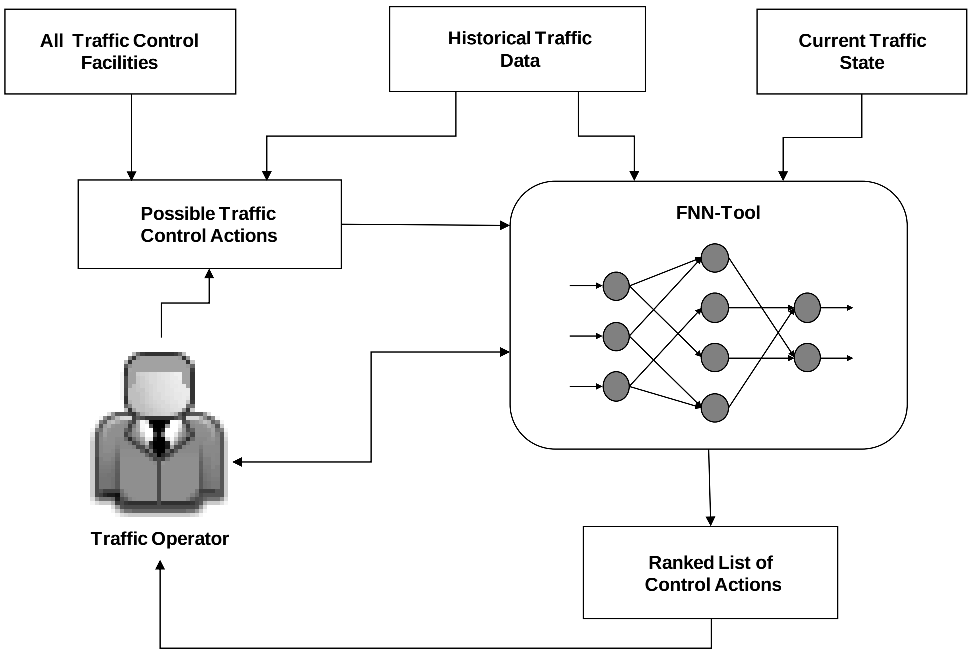

All Traffic Control Facilities
Historical Traffic Data
Current Traffic State
Possible Traffic Control Actions
FNN-Tool
Traffic Operator
Ranked List of Control Actions

Figure 4.1: Structure of Intelligent Traffic Control Decision Support System (ITC-DSS).

The current traffic state is characterized for each section of the network by: the average traffic state (including, e.g., densities, flow, speeds), the average boundary conditions (including demands at the origin links and the output restrictions at the destination links), and the incident status (including capacity reduction caused by the incident and the duration of the incident). Given the current traffic state and a set of possible control actions S for the given road network, the ITC-DSS employs a pre-trained Fuzzy Neural Network Tool (FNN-Tool) (see the next section for details) to predict the performance of each ca_i for the current traffic state. The inputs of the FNN-Tool are the values of the current traffic state and control action ca_i and the output of the FNN-Tool is the performance of ca_i . Once the performance of all control actions in S are evaluated using the FNN-Tool, S is ranked based on the control actions aggregated performance and provided to the operator in real-time. The pseudo-code shown in Figure 4.2 summarizes the main process of the proposed ITC-DSS:

```
Create $S = \{ca_1, ca_2, ca_3, \dots, ca_n\}$;
Set Traffic_State_Input = the current traffic state;
For ( $i$ =1; $i \leq n$; $i$++) do
 {
   Control_Input =  $ca_i$;
   $ca_i$_Performance =
      Execute FNN-Tool (Traffic_State_Input, Control_Input);
   $ca_i$_Agg_Pefromance = Cal_Agg_Performance($ca_i$_Performance);
 }
Output_Ranked_List= Rank($S$,$ca_i$_Agg_Pefromance) ;
```

Figure 4.2: The main process of the proposed ITC-DSS

4.3.2 Aggregated Performance

There are a range of traffic performance criteria such as queue lengths at the origins of the network, total travel times, total distances travelled, number of vehicles entering the network, the number of vehicles leaving the network, etc [**57**].

These can be considered to assess the performance of a control action. The aggregated performance of each control action ca_i can be calculated by considering one or more of the performance criteria, or by using a weighted sum approach as shown by the following equation:

$$P^i = \frac{\sum_{d=1}^{N} w_{C_d} \; E_{C_d}^i}{\sum_{d=1}^{N} w_{C_d}} \tag{4.1}$$

where $0 \leq P^i \leq 1$ represents the aggregated performance of control action ca_i for the given traffic state; w_{C_d} is the weight of the performance criterion C_d; and *N* is the number of performance criteria considered. These weights (w_{C_d}) are usually selected by the operators based on current traffic management policies and other considerations; $E_{C_d}^i$ is the evaluation of control action ca_i over the performance criterion C_d for the given traffic state ($E_{C_d}^i$ is in the range [0,1], where a low value of $E_{C_d}^i$ indicates a low performance of ca_i over the performance criterion C_d). $E_{C_d}^i$ is calculated for control action ca_i as follow:

$$E_{C_d}^i = 1 - \left(\frac{C_d - C_d^{min}}{C_d^{max} - C_d^{min}} \right) \tag{4.2}$$

where C_d^{min} and C_d^{max} are the minimum and the maximum values of C_d.

To clarify exactly how the aggregated performance of control action ca_i is calculated, consider the following example.

Assuming that the control action ca_i has the following performance (as obtained from the FNN-Tool – detailed in the next section):

– Total Travel Time (TTT) = 4484.64 [veh*h]

– Total Distance Travelled (TDT) = 173100.2 [km]

Now calculating $E_{C_d}^{i}$ for each performance criterion C_d using Equation 4.2:

$$E_{TTT} = 1 - \frac{4484.64 - 3000}{10000 - 3000} = 0.79$$

$$E_{TDT} = 1 - \frac{173100.2 - 80000}{250000 - 80000} = 0.45$$

Let, w_{TTT} and w_{TDT} be 1.5 and 0.5 respectively.

Thus:

$$P^{i} = \frac{(0.79 * 1.5) + (0.45 * 0.5)}{1.5 + 0.5} = 0.70$$

In this example, TTT is assigned a larger weight ($w_{TTT} = 1.25$) than TDT ($w_{TDT} = 0.25$), indicating the importance of that performance criteria at that time. w_{C_d} , C_d^{min} and C_d^{max} are adjustable factors and can be changed on-line by the user (i.e. the operator in the traffic control centre) depending on the current situation (e.g. C_d^{min} and C_d^{max} can be the desired and the worst values of C_d, respectively, at that period). This gives the proposed system ITC-DSS more flexibility to deal with different traffic situations and different traffic management policies. Furthermore, this increases the reaction between the system and the operator which increases the acceptance of the proposed system by the traffic operators.

4.3.3 Fuzzy Neural Network Tool (FNN-Tool) Structure

In this section, we describe the structure and the function of the proposed FNN-Tool. The learning process and the performance analysis of the FNN-Tool will be discussed in detail in Chapters 5 and 6. The FNN-Tool used in the study is of the Mamdani type and its structure is similar to the structure considered in [90] and [134]. It is a multilayer neural network-based fuzzy system - the topology is shown in Figure 4.3. The FNN-Tool has a total of five layers: the input layer; the condition layer; the fuzzy-rules layer; the consequence layer; the output layer. Each layer performs an operation for building the fuzzy system. The input and the output layers are represented as vectors $X^n = [x_1, x_2, \dots, x_{n1}]$ and $Y^n = [y_1, y_2, \dots, y_{n2}]$, where n_1 and n_2 represent the number of the input and output non-fuzzy variables. The process of each layer is as described below:

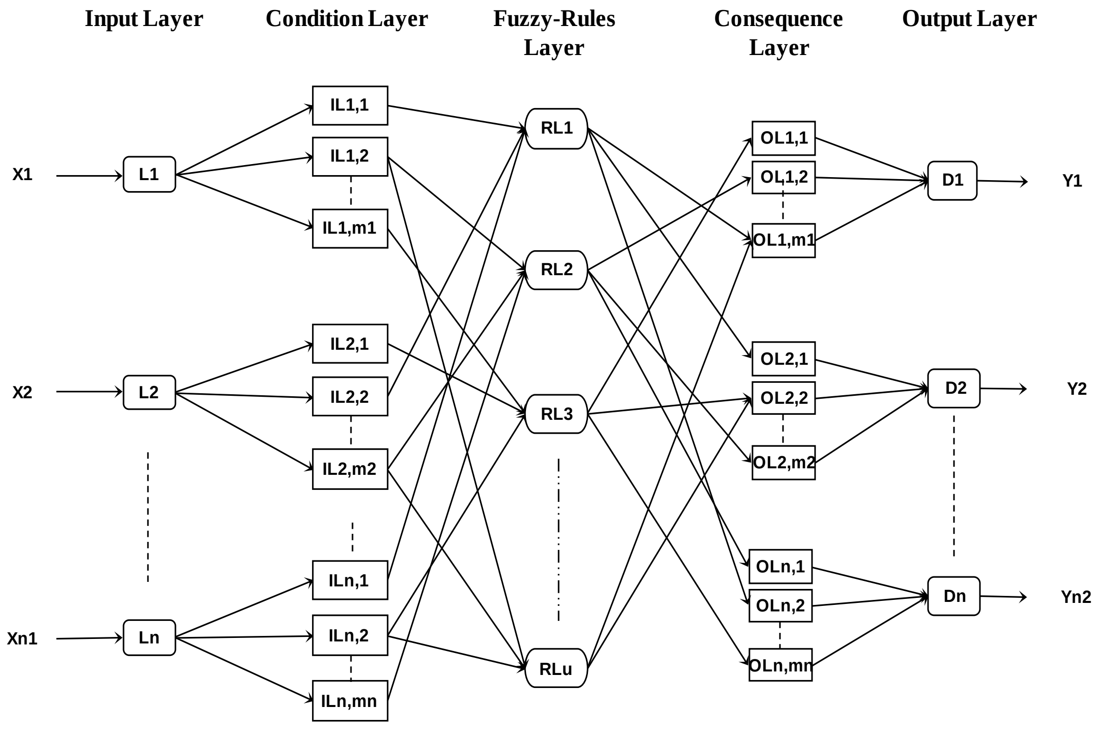


Figure 4.3: Structure of the Fuzzy Neural Network-Tool (FNN-Tool).

Layer 1(Input Layer): nodes at this layer are input nodes which represent input linguistic variables such as “age”, “weight”, and “speed” and directly transmit non-fuzzy input values to the next layer. Each node in this layer is connected to only those nodes of Layer 2, which represent the linguistic values of corresponding linguistic variables. In our case, the input linguistic variables are the components of the current traffic state (e.g. traffic densities, flows, speeds, inflow demands, outflow restrictions, incidents status, etc.) and control action ca_i.The link weights, $W_{n,m}$, between this layer and the next layer is unity. The output $o_n^{(1)}$ of this layer is given as follows:

$$o_n^{(1)} = x_{n1} \tag{4.3}$$

where x_{n1} is the input of the input neuron L_n in layer 1.

Layer 2 (Condition Layer): this layer defines the fuzzy sets and membership functions for each of the input factors. Nodes in this layer act as a membership function and represent the terms of the respective linguistic variable, such as “low”, “medium”, or “high”. The input values are fed to this layer that calculates the membership degree. In our model this is implemented using the Gaussian membership function (see Figure 4.4). The main reason behind the use of the Gaussian membership function instead of triangular or trapezoidal functions is to ensure differentiability, as required by the back-propagation algorithm employed in the last stage of the learning process [134]. The connection weights in this layer are unity. The output $o_{n,m}^{(2)}$ of input-label node $IL_{n,m}^{(2)}$ is given as follows:

$$o_{n,m}^{(2)} = e^{-\frac{(o_n^{(1)} - c_{n,m}^{(2)})^2}{(\sigma_{n,m}^{(2)})^2}} \tag{4.4}$$

where $c_{n,m}^{(2)}$ and $\sigma_{n,m}^{(2)}$ are the centres (or means) and the widths (or variances) of the membership function for the input-label node $IL_{n,m}^{(2)}$ respectively, where $IL_{n,m}$ denotes the *m*th input label of the linguistic node *n*.

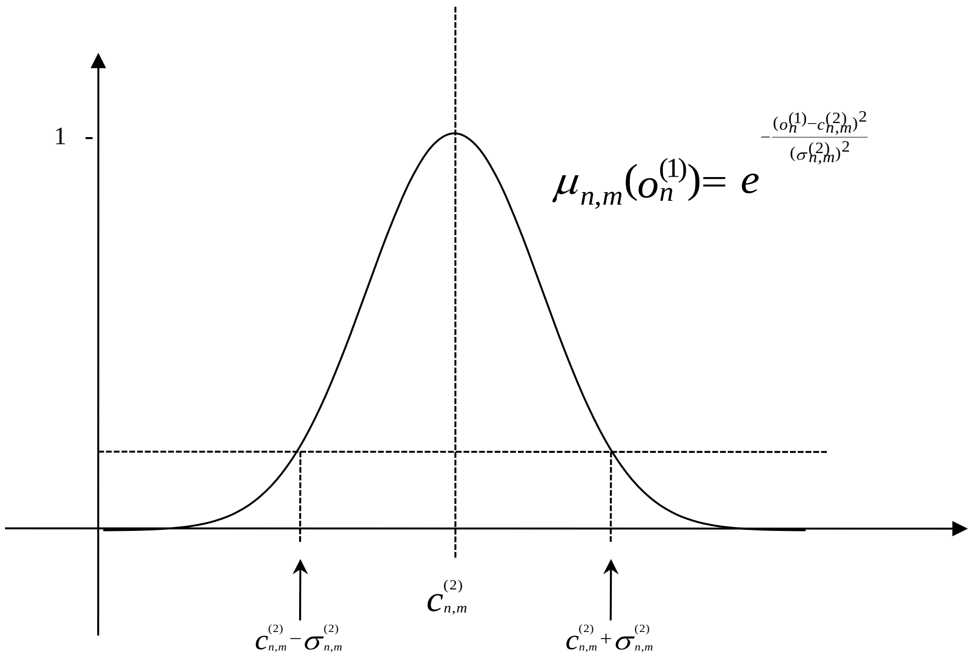

1
$\mu_{n,m}(o_n^{(1)}) = e^{-\frac{(o_n^{(1)} - c_{n,m}^{(2)})^2}{(\sigma_{n,m}^{(2)})^2}}$
$c_{n,m}^{(2)} - \sigma_{n,m}^{(2)}$
$c_{n,m}^{(2)}$
$c_{n,m}^{(2)} + \sigma_{n,m}^{(2)}$

Figure 4.4: Gaussian membership function.

Layer 3 (Fuzzy-Rules Layer): this layer defines all possible fuzzy rules to specify qualitatively how the output parameter is determined for various instances of the input parameters. Each node in this layer represents a fuzzy rule such as “if the Traffic Density is low AND the Control Action is Ramp Metering, then the Total Travel

Time is Medium and the Total Distance Travelled is Low”. The nodes in this layer perform the AND operation. The output $o_u^{(3)}$ of a rule node RL_u at the layer 3 is given as follows:

$$o_u^{(3)} = \min_{i \epsilon U} \left(o_i^{(2)}\right) \tag{4.5}$$

where U is the set of indices of the nodes in layer 2 that are connected to node RL_u in layer 3.

Layer 4 (Consequence Layer): each node in the consequence layer represents a possible consequent part of a fuzzy rule (such as “low” and “high”). The connection weights $W_{u,nm}$ of the links connecting nodes RL_u in Layer 3 to $OL_{n,m}$ in layer 4 represent certainty factors (CFs) of the corresponding fuzzy rules when inferring fuzzy output values. Each node of this layer performs the fuzzy *OR* operation to integrate the field rules leading to the same output linguistic variables. The initial values $W_{u,nm}$ are set to unity. The output $o_{n,m}^{(4)}$ of a consequence node $OL_{n,m}$ in Layer 4 is given as follows:

$$o_{n,m}^{(4)} = \max_{u \epsilon G} \left(o_u^{(3)} \; W_{u,nm}\right) \tag{4.6}$$

where G is the set of indices of the nodes RL_u in Layer 3 that are connected to node $OL_{n,m}$ in Layer 4.

Layer 5 (Output Layer): this layer is the defuzzification layer, where each node at this layer represents a single output variable. In our case, the output variables are the performance criteria (e.g. the queue lengths at the origins of the network, total travel times, total distance travelled, number of vehicles entering the network, number of vehicles leaving the network, etc.). In this layer, either the Centre of Gravity (COG) or Centre of Area (COA) method can be used to compute a crisp output signal for each node. In the ITC-DSS, we use COG; the output $y_n^{(5)}$ of an output node D_n in Layer 5 is given as follows:

$$y_n^{(5)} = \frac{\sum_{k \epsilon H}\left(o_{u,nm}^{(4)} \times c_{u,nm}^{(4)} \times \sigma_{u,nm}^{(4)}\right)}{\sum_{u \epsilon H}\left(o_{u,nm}^{(4)} \times \sigma_{u,nm}^{(4)}\right)} \tag{4.7}$$

where H is the set of indices of the nodes $OL_{n,m}$ in Layer 4 which are connected to node D_n in Layer 5 and $c_{u,nm}^{(4)}$ and $\sigma_{u,nm}^{(4)}$ are respectively, the centre and width of the membership function of the output linguistic value represented by $OL_{n,m}$ in Layer 4. The weights of links from the nodes in Layer 4 to the nodes in Layer 5 are unity. Therefore, only the learnable weights in the FNN-Tool network are $W_{u,nm}$ between Layers 3 and 4.

4.4 Application of ITC-DSS

In this section the technical feasibility of ITC-DSS is assessed using a road traffic case-study of a part of the traffic network of the city of Riyadh in Saudi Arabia.

4.4.1 Riyadh Traffic Case Study

The selected sub-network, which is shown in Figure 4.5, is one of the busiest parts of the Riyadh network, because it is used mostly for traffic approaching the city centre. This sub-network includes 10-km of King Fahad highway with four lanes each way. Two main roads (*R1* and *R2*) separate from King Fahad highway and run parallel to it and then join it again. Traffic enters the section from two origins (*O1* and *O2*) and leaves it through two destinations (*D1* and *D2*). In this case study, we only consider traffic going from south to north (i.e. towards the city centre).

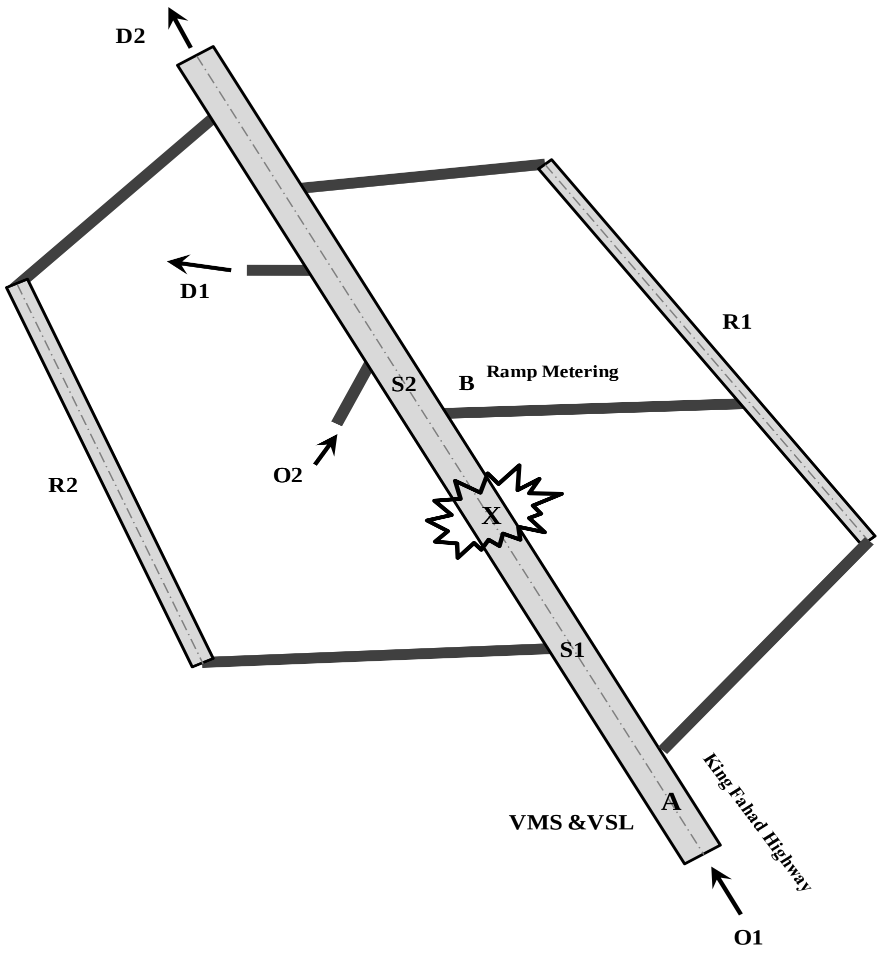


Figure 4.5: The sub-network considered in the traffic case study.

4.4.2 Case Study Design and Testing

The aim of this section is to illustrate how the proposed ITC-DSS works. Only a limited number of inputs, control actions, and training data have been considered at this stage. However, an increase in the number of inputs, possible control actions, and training data should not affect the validity of the proposed system. We have considered the following variables in our case study:

Three traffic factors to represent the traffic state:

- Average traffic demand (*TDm*): the average traffic inflows at the origin links of the sub-network (*O1* & *O2*).
- Average traffic density (*TDn*): the average number of vehicles per km per lane on the King Fahad highway.
- Incidents status (*IS*): only the severity of incidents is considered in this case study.

We assume that the traffic flow in *R1* and *R2* is very smooth.

Five traffic control actions:

- ca_1: Using VMS at point *A* to direct traffic that goes to *D2* to use road *R1*.
- ca_2: Using VMS at point *A* to direct traffic that goes to *D1* to use road *R1*.
- ca_3: Using VMS at point *A* to direct traffic that goes to *D1* and traffic that goes to *D2* to use road R1.
- ca_4: Using VMS at point *A* to direct traffic that goes to *D2* to use road *R2*.
- ca_5: Using VMS at point A to direct traffic that goes to *D1* to use road *R1* and on Ramp Metering at point *B*.

Two evaluation criteria for calculating the aggregated performance of the control actions are as follows:

- Total Travel Time (*TTT*):

  (w_{TTT} = 1.5 ; TTT_{min} = 3000 ; TTT_{max} = 10000).

- Total Distance Travelled (*TDT*):

  (w_{TDT} = 0.5 ; TDT_{min} =80000; TDT_{max} = 250000).

Thus, the inputs of FNN-Tool will be X^4= [*TDm*, *TDn*, *IS*, ca_i], where $i \in \{1, 2, 3,4,5\}$, and the outputs will be Y^2=[TTT_i, TDT_i]. The data needed for the training of the FNN-Tool has been generated using the traffic simulation model METANET. The learning process of the FNN-Tool as well as the process of generating traffic data by METANET, are described in Chapter 6.

For the purpose of using METANET, we assume the following traffic splitting rates:

Traffic entering King Fahad highway is divided as follows:

- 29% goes to destination *D1* (97% use King Fahad highway, and 3% use road *R1*)
- 71% goes to destination *D2* (90% use King Fahad highway, 6% road *R2*, and 4% use road *R1*)

In order to apply the proposed ITC-DSS, it has been employed to rank the five control actions ($ca_1, ca_2, ca_3, ca_4,$ and ca_5) based on their aggregated performance P^i on the following traffic state:

– *TDm* = 5500.

– *TDn* = 32.

– *IS* = 75%.

Table 4.1 illustrates the final results of this case study. The predicted TTT and TDT of the five control actions on the traffic state under consideration obtained by ITC-DSS, are summarized in columns 2 and 4. For each control action, E_{TTT} and E_{TDT} have been calculated using Equation 4.2 and summarized in columns 3 and 5. The aggregated performance P^i of each control action has been calculated using Equation 4.1 and summarized in column 6. As can be seen, the ITC-DSS recommends ca_3 as the most suitable solution to control the traffic state under consideration with aggregated performance (0.81), while it does not recommend ca_2 with aggregated performance (0.52). In order to evaluate the performance of the proposed ITC-DSS, we have also make a comparison between the results of the ITC-DSS and the results obtained by the traffic simulation model METANET.

Control Actions (ranked)	TTT	E_{TTT}	TDT	E_{TDT}	P^i
ca_3	3101.56	0.99	201913.5	0.28	0.81
ca_1	4484.64	0.79	173100.2	0.45	0.70
ca_4	5483.23	0.65	166845.7	0.49	0.61
ca_5	7007.4	0.43	111118.8	0.82	0.53
ca_2	7013.02	0.43	111606.7	0.81	0.52

Table 4.1: The performance evaluation of the control actions ca_1, ca_2, ca_3, ca_4, and ca_5 on the selected traffic state.

Generally, the results obtained from this case study show that the proposed ITC-DSS can be effectively used to quickly and roughly rank several control actions according to their aggregated performance. Moreover, the comparison between ITC-DSS and METANET illustrates that ITC-DSS indicates the same trend as the simulation model, which confirms the validity of ITC-DSS in predicting the control action performance. However, the time ITC-DSS needed to calculate the performance of a given control action is much less than the time needed by the simulation model. For example, METANET needs about 1 minute to evaluate one control action, while our proposed system can evaluate and rank five different traffic control actions in under 1 second.

Since the proposed system has been developed using a trained FNN-Tool, the main advantage of the proposed system is its speed of execution. Also, the proposed system allows the user to evaluate a set of control actions in one process, unlike their being evaluated one by one as in the simulation model.

4.5 Summary

Identifying the most promising control action among a large number of possible control actions to manage traffic congestion is a recurrent problem facing traffic control centres. This chapter has described an Intelligent Traffic Control Decision Support System (ITC-DSS) for traffic control centres to help the operator to identify the most suitable control action. The ITC-DSS uses a pre-trained fuzzy neural network (FNN-Tool) to predict the performance of several control actions. This gives a ranked list of control actions, of which the best-scoring control actions can be applied directly or, for more investigation, they can be assessed in more detail via, e.g.

microscopic or macroscopic traffic simulations.

In order to evaluate and test the proposed ITC-DSS, a case study of a section of the ring-road around Riyadh has been presented and discussed. The results obtained from the ITC-DSS clearly demonstrate its suitability in terms of processing speed, accuracy and flexibility. The essential results of this chapter have been reported in [3].

In this chapter we have demonstrated the technical feasibility of the proposed ITC-DSS with a small-size network and for a limited number of traffic situations and control actions. In Chapter 7 we will present a major extension and improvement of ITC-DSS in order to obtain a scalable control system.

In the next chapter, a GA-based fuzzy rules identification method will be presented to be used later (in Chapter 6) for generating fuzzy rules in the learning process of the FNN-Tool.

Chapter 5

GA-Based Fuzzy Rules Identification Method for Fuzzy Neural Networks

5.1 Chapter Overview

In this chapter we present a GA-based fuzzy rules identification method for fuzzy neural networks. Using the known membership functions, the GA based method initially considers all possible rules then uses the training data and the fitness function to perform rule-selection. In the previous chapter we introduced the structure and the functions of the FNN-Tool. The GA-based method presented in this chapter represents the second stage of the proposed three stage-based learning approach of the FNN-Tool. The first and third stages are initializing the membership functions of both input and output variables by determining their centres and widths using a self-organizing algorithm, and fine tuning the derived structure and parameters using the back-propagation learning algorithm. Detailed descriptions of the three stage-based learning approach of the FNN-Tool including the first and third stages, is given in the next chapter. This chapter concentrates on the application of the proposed GA-based method for fuzzy rules identification. The next section of this chapter reviews rule identification in the fuzzy neural network framework. Section 5.3 discusses the proposed GA-based fuzzy rules identification method. The validation of the proposed GA-based method is tested in Section 5.4.

5.2 Fuzzy-Rule Identification

5.2.1 Rule-identification Methods

An important topic in designing a fuzzy neural network is the identification of fuzzy rules. However, there is limited research reported for systematic design procedures [134]. The main research aim in the identification of the fuzzy rules in fuzzy neural networks is to learn and modify the rules from past experience. Various methods have been used for the identification of the fuzzy rules in fuzzy neural networks. Quek and Zhou [136] have classified the rule identification methods into the following three categories:

First category: This category includes methods that use linguistic information from experts to identify fuzzy rules in the fuzzy neural networks prior to the application of neural network techniques to adjust the rules [24], [182], [22], [98], [101], [150], [194]. For example, the knowledge provided by pilot studies has been used to design a fuzzy-logic based multi-input/multi-output roll controller for the advanced technology wing in [24], operational instructions provided by experienced operators of plants have been used to build a neuro-fuzzy adaptive control strategy for refuse incineration plants in [98]. Students' characteristics elicited from their teachers have been used to design a neuro-fuzzy model to diagnose student behaviour for the purpose of adapting pedagogical decisions to the individual student in [150]. Although this type of approach converges faster during training and performs better, it is rather subjective since linguistic information from experts may vary from person to person, and from time to time. This type of approaches has been widely used in first generation expert systems.

Second category: This category includes methods that use numerical information to identify fuzzy rules in fuzzy neural networks prior to the application of neural network techniques to adjust the rules [104], [55], [50], [135], [4], [138]. In this type of approach, unsupervised learning algorithms such as self-organizing [95] and competitive learning algorithms [97] are widely used to identify fuzzy rules from the numerical training data. For example, in [104], a self-organized learning algorithm has been employed to automatically construct a neural-network-based fuzzy logic control and decision system by learning the training example itself. Ang and Quek [4] have employed an unsupervised learning algorithm (i.e. discrete incremental clustering) to compute the input fuzzy sets and the output fuzzy sets needed to create a rules structure for their proposed fuzzy neural network. In [135] Fuzzy Kohonen Partitioning (FKP) and the Pseudo Fuzzy Kohonen Partitioning clustering algorithms have been employed to structure the fuzzy neural networks technique. Initially, a cluster analysis is used on the numerical training data and then the fuzzy rules are generated through the proper connections of these computed clusters.

Since the numerical data is the only source of information in this type of approach, it must be representative. Otherwise, the derived fuzzy rules will be ill-defined. The GA-based approach presented in this chapter to identify fuzzy rules falls into this category.

Third category: This category of approaches is using for supervised learning algorithms (particularly the backpropagation technique) to identify the fuzzy rules in the fuzzy neural networks[145], [75], [107], [184], [144]. These fuzzy neural networks are basically multilayered, with the inputs and outputs as fuzzy membership values that satisfy certain constraints. The backpropagation learning algorithm is often utilized in such fuzzy neural networks to produce the mapping from inputs to outputs. In this approach the fuzzy neural network appears as a black box at the end of the training process. Shann and Fu [145] have proposed a two phase learning procedure for the FNN. The first phase is an errorbackprop (EBP) training phase, and the second phase is a rule-pruning phase. The EBP learning algorithm is based on a gradient descent search in the network, in which some of the node functions are formulated with competitive operations such as min and max operators. The winners of these competitive operators are determined in the forward pass of the learning algorithm. Ishibuchi et al. [75] used interval vectors to represent fuzzy inputs and outputs in a fuzzy multilayer perceptron (MLP). The backpropagation algorithm was applied to generate different fuzzy IF–THEN rules from a few sample rules (used during training). Yin et al. [184] have developed a fuzzy neural model to predict traffic flow in an urban street network. They used a fuzzy approach to classify the input vectors data into a number of clusters, then employed a single layer neural network to specify the input-output relationship (rules) between the vectors.

5.2.2 Genetic Fuzzy System

Recently, the use of GA in the design of fuzzy systems (including rules identification, membership function adjustment, and training fuzzy rule-based models to represent specific data) has seen a considerable research effort compared with other approaches. This initiative has led to the coining of the term Genetic Fuzzy System (GFS), which is, essentially, a fuzzy system with a learning process based on GA [26]. The strong searching capacity of GA has been utilized in GFS for: (i) determining (or adjusting) membership functions with a fixed number of fuzzy rules [86], [123]; (ii) finding fuzzy rules with a known membership [19], [64], [46]; (iii) finding both membership functions and fuzzy rules simultaneously [66], [179], [76],; (iv) adjusting (or optimizing) fuzzy system parameters [155], [168], [169].

Karr [86], who has employed GA to train membership functions, was the pioneer. Nomura et al.[123] have determined the number of fuzzy sets and the membership function of each fuzzy set using GAs. On the other hand, Hoffmann [64] has proposed a learning method for fuzzy rule-based systems using the iterative rule learning approach. The fuzzy rule base is constructed in an incremental fashion, where GA optimizes one fuzzy classifier rule at a time. Another use of GA with the iterative rule learning approach has been reported in [46].

Using GA for finding both membership functions and fuzzy rules simultaneously, is a very common approach. Wu and Liu [179] have proposed a GA-based approach for the simultaneous design of membership functions and fuzzy control rules. In their proposed approach, the triangular membership function variables (the left and right widths and the locations of their peaks) and all possible fuzzy rules have been transformed into real-coded chromosomes to be optimized by GA. Ishibuchi and Yamamoto [76] have proposed a genetic algorithm-based approach for pattern classification problems consisting of two phases: candidate rule generation by rule evaluation measures in data mining, and rule selection by multi-objective evolutionary algorithms. In the rule selection phase, they have used three objective functions simultaneously: the maximization of the classification accuracy, the minimization of the number of selected rules, and the minimization of the total rule length.

GA has not only been employed to tune membership functions, but it has also been used to optimize the architecture of a fuzzy neural network. Wang et al. [168] proposed a GA-based approach for a feedback direct adaptive fuzzy-neural controller to tune the online weighting factors. In particular, they have used a reduced-form genetic algorithm to adjust the weightings of the fuzzy-neural network. Su et al. [155] have developed a fuzzy neural network controller to create suitable conditions for crop growth. GA has been used to train the architecture of their proposed fuzzy neural network controller. Wang et al. [169] have proposed a simplified GA to adjust both control points of B-spline membership functions and the weights of fuzzy-neural networks.

Starting with all possible fuzzy rules then using the GA to identify the relevant rules is a very promising approach. This approach was adopted by Castro & Camagro [19] who proposed a three stage-based approach for identifying fuzzy rules from numerical data. The three stages are: a feature selection process, a GA for deriving fuzzy rules and, finally, another GA for optimizing the rule base. The main disadvantage of this approach is that, depending on the number of input-output variables and the number of their fuzzy sets, the total number of possible rules can be extremely large, making it difficult to encode and generate the chromosomes. Consequently, the learning process can become overloaded. However, starting the GA process with all possible rules (if possible) is still preferred because it decreases the chance of missing any relevant rule which, in turn, minimizes the final error.

In this chapter we present a GA-based method for identifying fuzzy rules from numerical data using previously defined and fixed fuzzy sets. The main advantage of our GA-based method is that GA is used only for identifying fuzzy rules. Using GA to optimize fuzzy membership functions and fuzzy rules simultaneously (such as in [66] and [76]), which is avoided in our proposed method, makes the GA suffer from the curse of dimensionality, because every fuzzy rule represents a different subspace of the input variables. Another advantage of the proposed GA-based method is that the fine tuning of the fuzzy rules weight is done in a separate learning stage (stage three). Consequently, integer representation (encoding) of the problem is used which reduces the length of the chromosome as well as making the size of the GA search space very small compared with other approaches (such as [64] and [179]) where the floating point numbers representation is implemented.

5.3 The Proposed GA-based Fuzzy Rule Generating Algorithm

5.3.1 GA-based Method

The proposed GA-based method is performed in the second stage to identify the fuzzy rules that are supported by a set of training data. A simple example of FNN with two input linguistic variables x_1 and x_2 and one output linguistic variable y is considered here to explain the design process of the presented GA-based learning approach. The self-organization learning algorithm in the first stage assigns each linguistic variable a number of fuzzy sets. Let us assume that we have three fuzzy sets {low (L), medium (M), high (H)}. Then the proposed GA-based method considers all possible rules for given fuzzy sets, as shown in the top part of Figure 5.1 (a). In this simple example, there will be a total of twenty seven possible rules. In fact these rules are made up of nine possible antecedents (preconditions) of fuzzy rules represented by nodes *RL1* ... *RL9* in the Fuzzy-Rules Layer. Each antecedent has links with three possible decision fuzzy sets (nodes in Consequence Layer: L, M and H). For example, the three possible fuzzy rules associated with node *RL1* are:

If x1 is L and x2 is L, then y is L.

If x1 is L and x2 is L, then y is M.

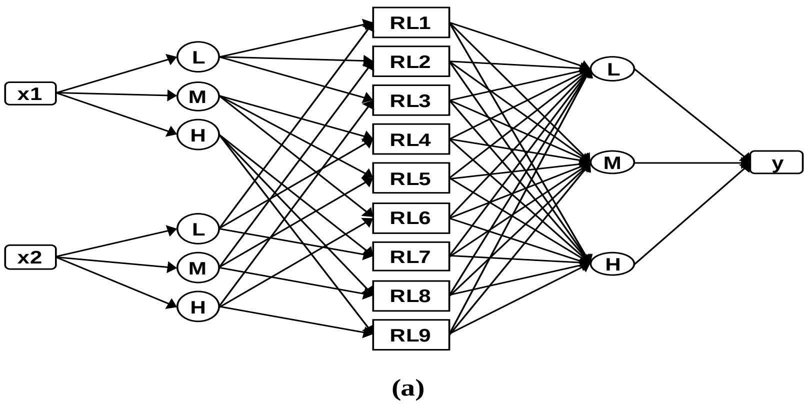
If x1 is L and x2 is L, then y is H.

In this way the total number of rules includes all possible fuzzy rules associated with all nodes. However, at most, only one of these three fuzzy rules can be used for making decisions. We used a GA-based learning approach to identify only the appropriate and relevant fuzzy rules by filtering out all other redundant rules.

A number of decisions must be made in order to implement the GA for generating appropriate fuzzy rules. In general there are five components that must be taken into account before using GA to solve a problem [20]:

- A genetic representation of solutions to the problem.
- A way to create an initial population of solutions.

- An evolution function which gives the fitness of each chromosome.
- Genetic operators that alter the genetic composition of offspring during reproduction (e.g. replacement, selection, crossover, mutation, etc)
- Values for the parameters that the GA uses (e.g. population size, probabilities of applying genetic operators, stopping criterion, etc)



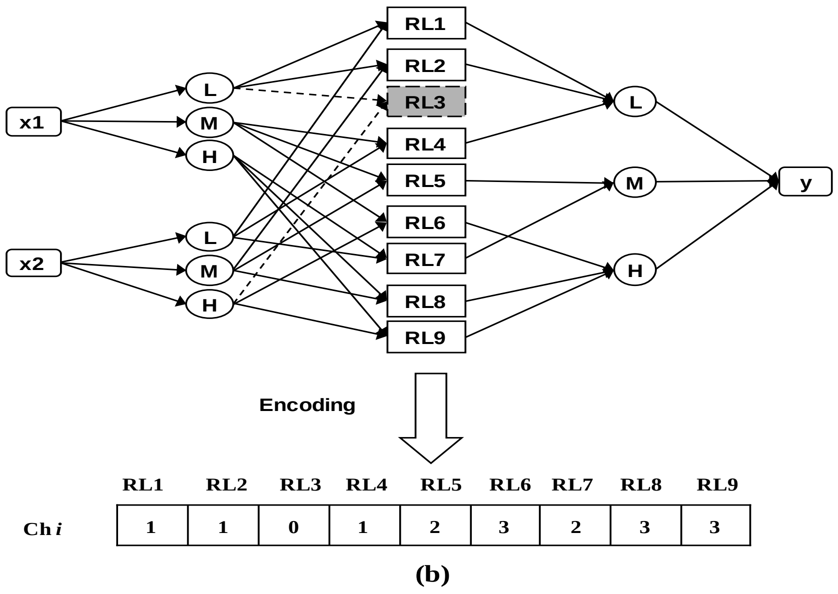


Figure 5.1: (a) All possible rules with two inputs and one output; (b) an example of encoding a chromosome.

5.3.2 GA Design

The following steps are employed for designing our method to identify appropriate fuzzy rules using the GA.

Encoding: the first step in the implementation of the GA technique is the encoding of the problem using an appropriate representation. The encoding is used to represent chromosomes (solutions), and to define the size and the structure of the search space. Here we propose integer strings as chromosomes to represent candidate solutions of the problem. The string is given by $(g_1, g_2, \dots, g_i, \dots, g_n)$, where g_i is an integer ($0 \le g_i \le m$) which indicates the link of the fuzzy nodes RL_i (i.e. nodes in the Fuzzy-Rules Layer) with the output nodes (i.e. nodes in the Consequence Layer); *n* is the number of nodes in the Fuzzy-Rules Layer; and *m* is the number of neurons in the Consequence Layer. In our example, the chromosome Ch_i has nine integers representing g_i, and $0 \le g_i \le 3$. The situation with $g_i = 0$ indicates there is no link between RL_i and nodes in Consequence Layer; $g_i = 1$ indicates that there is a link with 'L' node in Consequence Layer and so on. An example of encoding/decoding a chromosome is shown in Figure 5.1 (b).

Fitness function: in this step the goodness of every chromosome is evaluated by using a fitness function. The GA only needs a fitness value assigned to each chromosome. In this paper, we use a set of training data to calculate the fitness of each chromosome based on the following error function:

$$FIT = 1 - \left(\frac{1}{n_d} \sum_{i=1}^{n_d} (y_i - \hat{y}_i)^2 \right) \tag{5.1}$$

where n_d is the number of data, y_i is the *i*th actual output, and $\hat{y}_i$ is the *i*th model output. Equation 5.1 represents the sum of Root Mean Squares (RMS) of errors, between actual outputs and model outputs. The GA aims to maximize this fitness function in order to minimize the error value. This error value is dependent on the selected fuzzy rules.

GA operators: Figure 5.2 shows a pseudo-code which represents the working principle of a simple GA. Based on a number of experiments, we have selected GA operators and their parameters to be used for this application. The results of those experiments will be given in the next chapter. The GA operators used are the tournament selection [45], the elitist generation replacement, standard two-point crossover and a random mutation. The tournament selection method picks a subset of solutions at random from the population to form a tournament selection pool from which two solutions are selected with probability based upon the fitness values of the solutions. The elitist approach ensures that the best solution in the population pool is always retained.

The two-point crossover operator splits the selected solutions at two randomly chosen positions and exchanges the centre sections with a crossover probability. An example is illustrated in Figure 5.3 (a). The mutation operator changes the integer at each position in the solution Ch_i within the allowed range (i.e. $0 \leq g_i \leq m$) with a defined mutation probability. An example is illustrated in Figure 5.3 (b). The initial population of chromosomes is created randomly. The stopping criterion for a GA run is to achieve the pre-specified error level.

```
Genetic_Algorithm ()
{
  formulate initial population;
  randomly initialize population;
  repeat
     evaluate objective function;
     find fitness function;
     apply genetic operators
           selection;
           crossover;
           mutation;
     apply replacement approach;
  until stopping criteria
}
```

Figure 5.2: The working principle of a simple genetic algorithm.

When the GA learning process is completed (e.g. when a pre-specified error level is achieved), we choose the best GA chromosome. This best chromosome is decoded to get the structure of the FNN-Tool by keeping only the rules that are indicated by the chromosome. A gene in a GA string with $g_i \neq 0$ represents a fuzzy rule to be considered and with $g_i = 0$ to be ignored. The weight for all rules is assumed to be 1 at this stage. Then the error level (e) can be improved by using the back-propagation learning algorithm (stage three as described in the next chapter) to fine tune the rules weights. By doing so, we train the FNN-Tool with the relevant fuzzy rules only.

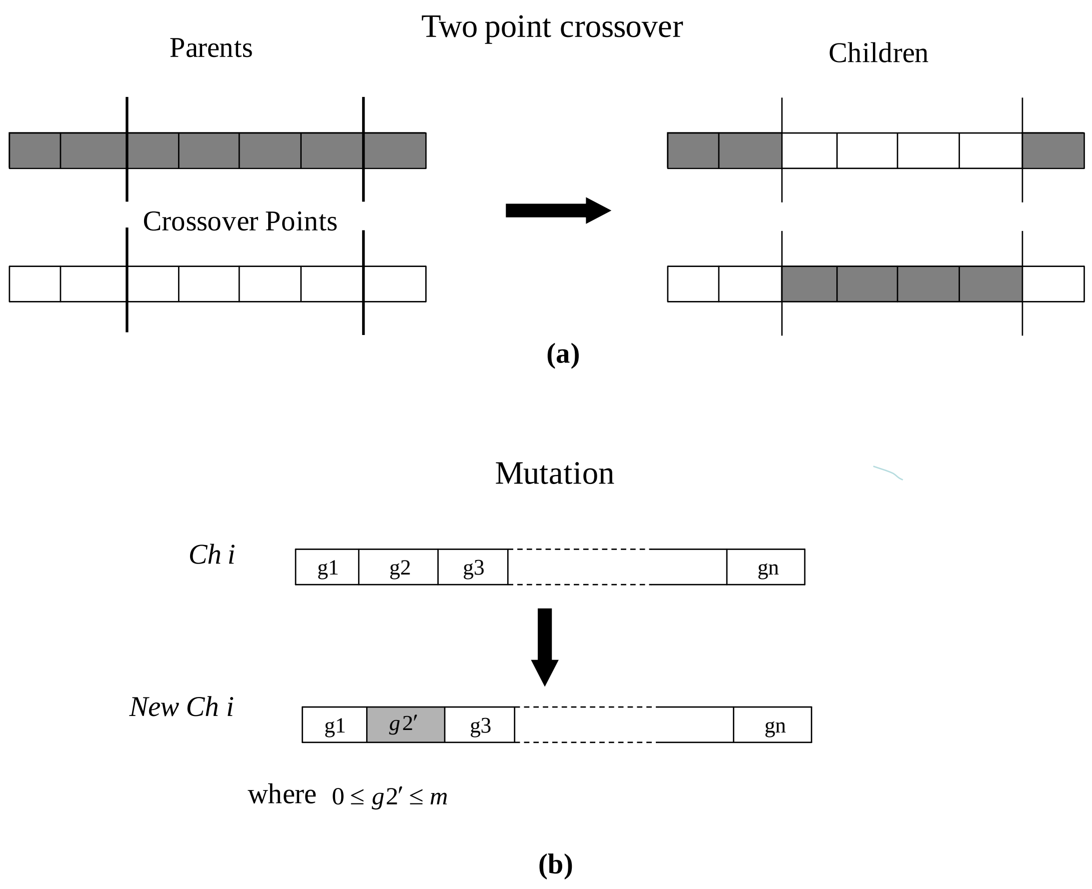


Figure 5.3: Considered GA operations (a) two point crossover; (b) mutation.

5.4 Validation Experiment

The main purpose of this chapter is to introduce the adopted learning approach for the fuzzy rules identification stage. In this section, we only test the validity of the GA-based learning approach for identifying the relevant fuzzy rules from all possible rules. The performance analysis of the GA-based learning approach is presented in the next chapter. In this validation experiment, the set of training data consists of a thousand data samples which are generated from 27 rules, some of which are listed below:

If x1 is low and x2 is low and x3 is low, then y is very low;

If x1 is low and x2 is medium and x3 is medium, then y is low;

If x1 is low and x2 is medium and x3 is high, then y is medium;

If x1 is medium and x2 is high and x3 is high, then y is high;

If x1 is high and x2 is high and x3 is high, then y is very high;

The weight (Degree of Coverage) of all rules is set to unity (i.e. $w_i = 1$).

From the above rules, it can be observed that there are three inputs and one output linguistic variables. For each input linguistic variable, its term set is defined as {low, medium, high}, while the output linguistic variable term set is defined as {very low, low, medium, high, very high}. Therefore, there are a total of 135 possible rules, and the resulting FNN consists of three linguistic nodes, nine input-label nodes (three for each linguistic node), twenty seven initial rule nodes, five output-label nodes, and one output node. In this case, the string (chromosome) is given by $(g_1, g_2, \dots, g_i, \dots, g_{27})$, where g_i is an integer ($0 \leq g_i \leq 5$). The GA-based approach is employed to identify the same 27 fuzzy rules from the 135 possible rules. The crossover and mutation probabilities considered in this experiment are 0.7 and 0.02 respectively. After a number of runs, the results show that the GA is able to quickly and correctly identify all 27 fuzzy rules with an error level equal to zero. Figure 5.4 presents an average performance graph of 20 experiments created by 100 generations of 20 chromosome populations.

This example demonstrates that the proposed GA-based method can identify correctly all relevant fuzzy rules in this small case. The evaluation of the effectiveness of the application of the proposed GA-based method is analyzed and presented in the next chapter, when it is employed for identifying the fuzzy rules of the FNN-Tool.

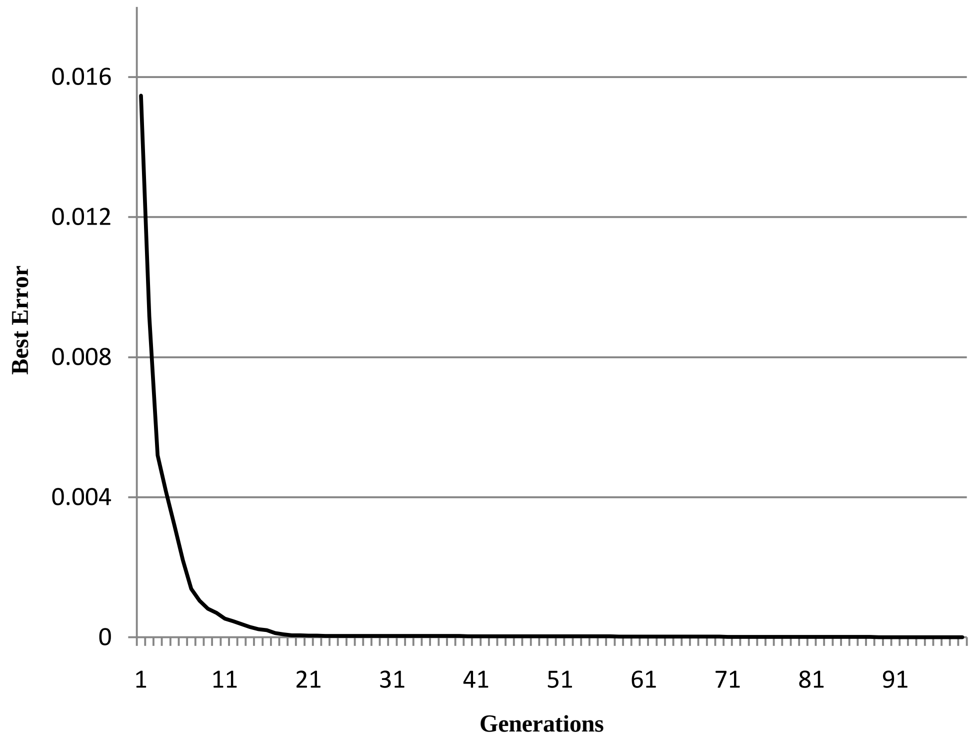


Figure 5.4 Performance graph for 100 generation of 20 chromosomes.

5.5 Summary

The identification of fuzzy rules in designing a fuzzy neural network is an important topic. This chapter has presented a GA-based method to identify the fuzzy rules for FNNs. The proposed GA-based method makes use of the known membership functions to identify only the relevant fuzzy rules. Initially, it has considered all possible rules and then used the training data and the fitness function to select a limited number of relevant fuzzy rules. The validity of the proposed method has been tested for the identification of all the relevant fuzzy rules. The results demonstrate the capabilities of the proposed GA-based method in terms of the quick and correct identification of the fuzzy rules.

As mentioned before, using the presented approach for identifying fuzzy rules with a large number of variables and fuzzy sets is not recommended because the total number of possible rules will be extremely large, which makes the genetic learning process very slow. In this case, instead of encoding the chromosomes with all possible rules, an alternative approach is to employ a clustering algorithm (e.g. [135]) just to identify an optimal number (n) of fuzzy rules to start with. In this case the encoding used to represent chromosomes is given by:

$$(h_1, g_1, h_2, g_2, \dots, h_i, g_i, \dots, h_n, g_n)$$

where h_i is an integer ($1 \leq h_i \leq N$), which indicates the link of the fuzzy nodes RL_i (i.e. nodes in the Fuzzy-Rules Layer) with the input nodes (i.e. nodes in the Condition Layer); g_i is an integer ($0 \leq g_i \leq m$) which indicates the link of the fuzzy nodes RL_i (i.e. nodes in the Fuzzy-Rules Layer) with the output nodes (i.e., nodes in the Consequence Layer); N is the number of all possible rules, while m is the number of neurons in the Consequence Layer.

The fuzzy rules identification method that has been presented in this chapter represents the second stage of our proposed three stage-based learning approach of our FNN-Tool. In the next chapter, the first and third stages of the proposed learning approach of the FNN-Tool are presented with a detailed description. The research results of this chapter have been reported in [1].

Chapter 6

Learning Approach for Fuzzy Neural Networks

6.1 Chapter Overview

In this chapter we present a three stage-based learning approach for our fuzzy neural network (i.e. FNN-Tool) presented in Chapter 4. In the first stage of the proposed three stage-based learning approach, the membership functions of both input and output variables are initialized by determining their centres and widths, using a self-organizing algorithm. The GA-based fuzzy rules identification method, which was presented in the previous chapter, is performed in the second stage to identify the fuzzy rules. In the last stage, the derived structure and parameters are fine tuned using the back-propagation learning algorithm. The capability of the proposed learning approach is tested with a well-examined benchmark example, and its strength is analyzed through a comparative study with other approaches. Moreover, an application of the pre-trained FNN-Tool with the proposed three stage-based learning approach is demonstrated with regard to the traffic case study introduced in Chapter 4.

This chapter is organized as follows. The next section reviews the related works published in the literature. Section 6.3 highlights the various stages in the design of the learning process of the FNN-Tool. The performance of the proposed FNN-Tool is tested in Section 6.4. Section 6.5 discusses the application of the proposed learning process for the traffic case study presented in Chapter 4.

6.2 FNN Learning Methods

Employing an effective learning process is a critical aspect of designing a fuzzy neural network, especially when the expert knowledge is not available. A wide variety of learning methods associated with fuzzy neural networks exist. In general, we can classify the existing fuzzy neural networks learning approaches into two types. The first type includes methods that use linguistic information provided by human beings to create the structure of the fuzzy neural network, while the tuning and the configuration of the parameters and structures is achieved by using neural network techniques such as [98], [150] and [101]. In this type, experts are required to provide a clear description of the membership functions and fuzzy rules used for the construction stage. As mentioned before, this type of learning is subjective, since linguistic information from experts may vary from person to person, and from time to time.

The second type includes methods that use numerical information to create the structure of a fuzzy neural network. The tuning and the configuration of the parameters and structures is achieved by using neural network techniques such as [137],

[90] and [162]. This type is similar to the previous one in terms of the tuning and the configuration of the parameters and structures. However, the initial set of parameters and the structure of the fuzzy neural network are derived from numerical information instead of linguistic information. This type of learning method is suitable for applications where experts who can provide an organized description of the system are unavailable. However, since the set of training data is the only source of information employed in this type of fuzzy neural network, it has to be representative of the system's behaviour. A brief survey of some well-known examples of fuzzy neural systems and their learning methods is provided in the rest of this section.

ANFIS by Jang [77] is a well known neuro-fuzzy system which implements a Sugeno-like fuzzy system [157] in a five-layered network structure. Backpropagation is used to learn the antecedent membership functions, while a least mean squares algorithm determines the coefficients of the linear combinations in the consequent of the rule. ANFIS uses differentiable functions instead of min and max functions. ANFIS adjusts only the membership functions of the antecedent and consequent parameters, thus the rule base must be known in advance.

Quek and Zhou [137] have designed a five-layered fuzzy neural network named Outer-Product-based Fuzzy Neural Network (POPFNN). The learning process of POPFNN consists of three phases: self-organization, POP learning, and supervised learning. A self-organizing algorithm is employed in the first phase to initialize the membership functions of both the input and output variables by determining their centres and widths. In the second phase, the POP algorithm is run in one pass to identify the fuzzy rules that are supported by the training set. The derived structure and parameters are then fine-tuned using the backpropagation algorithm. The

POPFNN approach has been subsequently applied to develop several FNN systems such as (POPFNN-AARS(S) [138] and POPFNN-CRI(S) [5]).

Kim and Kasabov [90] have proposed an adaptive neuro-fuzzy system called HyFIS (Hybrid Neural Fuzzy Inference System), for building and optimising fuzzy models. The learning procedure in HyFIS consists of two phases. In phase one, the rule finding phase, the method proposed by Wang and Mendel [170] for deriving fuzzy rules from desired input–output data pairs is used to find the linguistic fuzzy rules and the initial structure of the neural fuzzy system. In phase two, the rule tuning phase, a supervised learning scheme, based on gradient descent learning, is used to optimally adjust the MFs for the desired outputs.

GenSoFNN (Generic Self-organizing Fuzzy Neural Network) is another Five layered fuzzy neural network developed by Tung and Quek [162]. The GenSoFNN network uses a new clustering technique called Discrete Incremental Clustering (DIC) to enhance its noise tolerance capability against noisy/spurious training data sets. The learning process of GenSoFNN also consists of three phases, self-organizing, rule formulation and parameter learning. Firstly, in the self-organizing phase, the problem domain is modelled by performing a cluster analysis on the numerical training data, using DIC technique to compute the input–output clusters. Next, fuzzy rules are produced by connecting the appropriate input and output clusters during the rule-mapping process of the learning phase in the rule formulation phase. Finally, back-propagation learning algorithm is used to fine-tune the parameters of the GenSoFNN network, to achieve the desired output response.

The Falcon-ART is an FNN architecture with 5 layers developed by Lin and Lin [105]. Before the training, Falcon-ART has only the input and output layers to represent the input and output linguistic variables respectively. The hidden layers for the input and output term nodes and the fuzzy rules are created and begin to grow as the learning cycle progresses. Falcon-ART dynamically partitions the input–output data spaces into trapezoidal fuzzy sets, tunes the trapezoidal membership functions that represent the linguistic terms, and determines the proper network connections between the input–output clusters and the fuzzy rules through a mapping process, with all these performed in a single pass of the training data set. The fuzzy Adaptive Resonance Theory (fuzzy ART) is used to perform the fuzzy clustering of the input–output spaces into fuzzy hyper-boxes (hyper-cubes). Falcon-ART then dynamically determines the proper fuzzy rules by connecting the appropriate input and output clusters (input and output hyper-boxes) by using a mapping process. Subsequently, a back-propagation learning algorithm is used to adjust the input–output membership functions.

Our proposed three stage-based learning approach presented in this chapter falls under the second type of FNN learning approach where the expert knowledge is not available and only numerical information is used for contracting and learning the system. The main reason behind the application of GA in the second stage instead of a self-organizing algorithm (or competitive learning algorithm) to identify fuzzy rules, is that since the membership functions are determined prior to the identification of fuzzy rules, self-organizing or competitive learning becomes redundant, because the boundary of the clusters in the input and output spaces have already been predefined [134].

One of the main advantages of the proposed three stage-based learning approach comparing it with, for example, HyFIS [90] and ANFIS [77], is that the FNN model is built for a system without prior knowledge about the partitions input/output space and the number of fuzzy rules. Moreover, the task of GA in the proposed three stage-based learning approach is determining the relevant rules from a set of all possible rules (i.e. rule-selection) only, and the fine tuning rule weights process is implemented in a separate learning stage (i.e. stage three). Another advantage of our proposed approach is that it is fast compared with those approaches presented in Lin and Lee [104], where rule-selection is performed by a competitive learning using rule weights with the rule with the largest weight being selected. In Lin and Lee's approach, the learning process involves iterative training before the system comes to a stable state, because a large number of all possible fuzzy rules are useless.

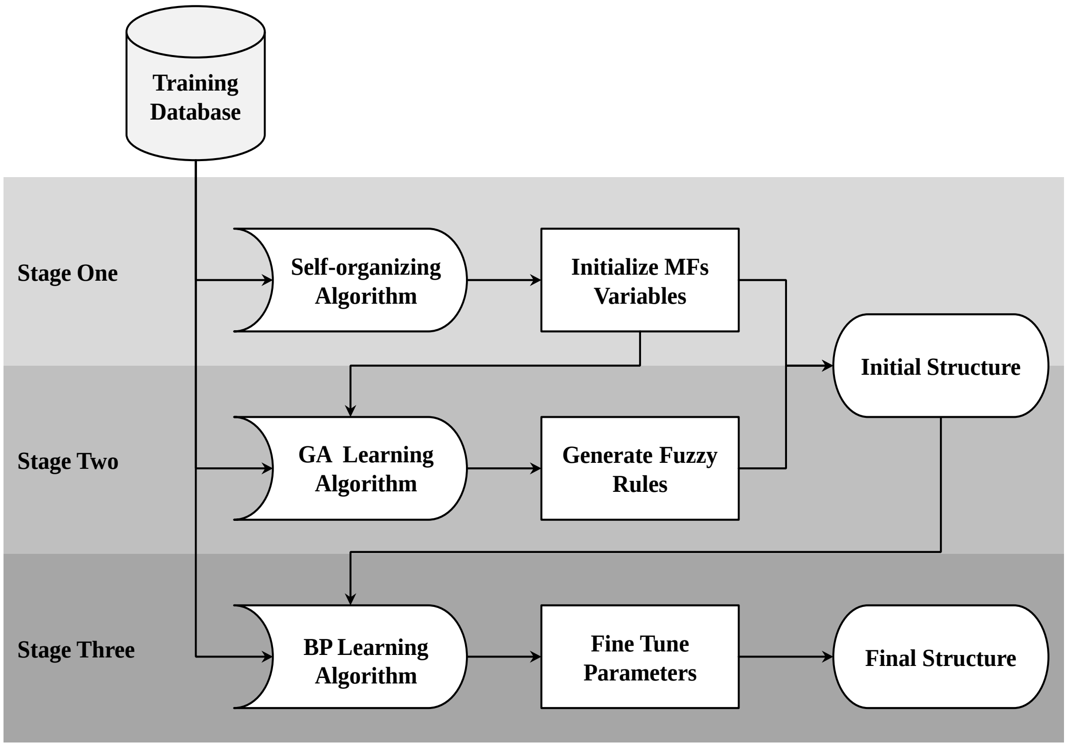


Figure 6.1: Flowchart of the proposed three stage-based learning approach.

6.3 The Proposed Learning Approach

The proposed learning approach for the FNN-Tool consists of three stages of learning as shown in Figure 6.1. The first stage (initializing input/output membership functions) and the second stage (identifying fuzzy rules) yield the initial structure of the FNN-Tool. The third stage (fine tuning the parameters) produces the final structure of the FNN-Tool. A detailed description of the three-stage learning approach is presented in the following sections.

6.3.1 Stage 1: Initialization with a Self- Organizing Algorithm

The first stage in the proposed learning approach initializes (self-organizes) the membership functions of both input and output variables of the FNN-Tool by determining their centres and widths. To perform this stage, we have employed a self-organizing algorithm. Alternatively, if expert knowledge is available, it can be used in this stage. Kohonen's feature-map algorithm [95], which has been employed in [138], is adopted in this work to identify the initial centres $c_{n,m}^{(2)}$ and $c_{k,nm}^{(4)}$ of the membership functions which represent the input and output label nodes $IL_{n,m}$ and $OL_{n,m}$. Kohonen's feature map algorithm , , is "the representation of knowledge in a particular category of things in general might assume the form of a feature map that is geometrically organized over the corresponding piece of the brain"[95]. Kohonen's algorithm is a self-organizing approach which takes a set of N-dimensional objects as inputs and produces a low-dimensional (typically two dimensional) grid, called a map.

The inputs and the output of the FNN-Tool are represented as vectors $X^n = [x_1, x_2, \dots, x_{n1}]$ and $Y^n = [y_1, y_2, \dots, y_{n2}]$, where n_1 *and* n_2 represent the number of the input and output variables. It must be noted that the input and output vectors are non-fuzzy vectors. That is, each element in X^n and Y^n has a non-fuzzy value. The self-organizing algorithm adopts the following steps:

Step1: initialize $c_{n,m}^{(2)}$ value:

$$c_{n,m}^{(2)}(T) = \min_{k \in X}(x_k) + \frac{1}{2}\left(\frac{m+1}{i}\right)(\max_{k \in X}(x_k) - \min_{k \in X}(x_k)) \tag{6.1}$$

where $m \in \{1,2,\dots,i\}$, i is the number of the membership functions that represent the terms of the respective linguistic, and variable x_k is the *k*th element of the input vector X^n.

Step2: find the closest:

$$\left\| x_k(T) - c_{n,closest}^{(2)}(T) \right\| = \min_{1 \le m \le i}\left(\left\| x_k(T) - c_{n,m}^{(2)}(T) \right\|\right) \tag{6.2}$$

where T is the training iteration.

Step3: Update $c_{n,m}^{(2)}$ value:

$$c_{n,m}^{(2)}(T+1) = \begin{cases} c_{n,m}^{(2)}(T) + \varepsilon(T)\left[x_k(T) - c_{n,m}^{(2)}(T)\right], \\ \qquad\qquad\qquad if\ c_{n,m}^{(2)}(T) = c_{n,closest}^{(2)}(T) \\ c_{n,m}^{(2)}(T), \qquad\qquad otherwise \end{cases} \tag{6.3}$$

where ε(T) is a monotonically decreasing scalar learning rate.

Step4: repeat steps 2 & 3 for T=T+1, while T< the limit on time iteration.

Step5: determine the width $\sigma_{n,m}^{(2)}$ value:

$$\sigma_{n,m}^{(2)} = \frac{\left|c_{n,m}^{(2)} - c_{n,closest}^{(2)}\right|}{2} \tag{6.4}$$

Similarly, the initial centres $c_{k,nm}^{(4)}$ and widths $\sigma_{k,nm}^{(4)}$ of the membership functions representing the output-label nodes, can be derived except that the output vector Y^n is used as the training data instead of the input vector X^n. It should be noted that the values of the centres and widths obtained here are all initial values which will be fine-tuned in the final stage using back-propagation learning. Since the aim of this stage is to reflect the rough locations of the clusters that are formed by the input and output data samples, any other appropriate clustering algorithm (e.g. K-means) can be used in this stage.

6.3.2 Stage 2: Rule Identification with GA-Based Method

The GA-based fuzzy rules identification method presented in the previous chapter is performed in the second stage to identify the fuzzy rules that are supported by a set of training data. As we mentioned before, the weights for all identified fuzzy rules are assumed to be 1 at this stage. A full description of the GA-based fuzzy rules identification method employed in this stage was given in Chapter 5. After completing this stage, the initial structure of the FNN-Tool is derived and should be ready for the last stage of the learning process.

6.3.3 Stage 3: Fine Tuning Stage with a Back-Propagation Learning Algorithm

After identifying the relevant fuzzy rules and the initial structure of the FNN-Tool, it can adjust its parameters using the back-propagation algorithm. The aim of this learning stage is to minimize the following error function:

$$E = \frac{1}{2}(y_i - \hat{y}_i)^2 \tag{6.5}$$

where y_i is the actual output and $\hat{y}_i$ is the model output for *i*th data.

From the structure of the FNN-Tool shown in Figure 4.2 (in Chapter 4), it can be observed that there are only five types of adjustable parameters. These are: centres $c_{n,m}^{(2)}$ and widths $\sigma_{n,m}^{(2)}$ of input-label membership functions, and centres $c_{u,nm}^{(4)}$, widths $\sigma_{k,nm}^{(4)}$ of output-label membership functions and the connection weights $W_{u,nm}$ of the links connecting nodes RL_u in layer 3 to $OL_{n,m}$ in Layer 4 (i.e. fuzzy rules weights).

Once an input training vector *x* is presented at the input layer during supervised learning, it is propagated forward through the neural network. Subsequently, an error signal is calculated and then feedback from Layer Five is used to adjust the parameters. The adjustment steps in the supervised learning for the FNN-Tool are stated as follows:

Layer 5-Output Layer: In this layer, the error signal $e_n^{(5)}$ is calculated using Equation 6.5 as follows:

$$e_n^{(5)} = -\frac{\partial E}{\partial y_n^{(5)}} = y_n - y_n^{(5)} \tag{6.6}$$

where y_i and $y_n^{(5)}$are the target and actual outputs of node *n* in Layer 5.

The adjusted centres $\Delta c_{u,nm}^{(4)}$ and widths $\Delta\sigma_{u,nm}^{(4)}$ of the output labels are calculated using Equation4.7 as follows:

$$\Delta c_{u,nm}^{(4)} = \frac{\partial E}{\partial y_n^{(5)}} \frac{\partial y_n^{(5)}}{\partial c_{u,nm}^{4}} = -(y_n - y_n^{(5)}) \frac{\sigma_{u,nm}^{(4)} \quad o_{n,m}^{(4)}}{\sum_m (\sigma_{u,nm}^{(4)} \quad o_{n,m}^{(4)})} \tag{6.7}$$

Hence, the $c_{u,nm}^{(4)}$ parameter is updated by:

$$c_{u,nm}^{(4)}(t+1) = c_{u,nm}^{(4)}(t) - \varphi \, \Delta c_{u,nm}^{(4)} \tag{6.8}$$

where φ>0 is the learning rate.

Similarly:

$$\Delta\sigma_{u,nm}^{(4)} = \frac{\partial E}{\partial y_n^{(5)}} \frac{\partial y_n^{(5)}}{\partial \sigma_{u,nm}^{(4)}} = \\ -\left(y_n - y_n^{(5)}\right) \frac{o_{n,m}^{(4)} \left(c_{u,nm}^{(4)} \left(\sum_m \sigma_{u,nm}^{(4)}\ o_{n,m}^{(4)}\right) - \left(\sum_m \sigma_{u,nm}^{(4)}\ o_{n,m}^{(4)}\ c_{u,nm}^{(4)}\right)\right)}{(\sum_m \sigma_{u,nm}^{(4)}\ \ o_{n,m}^{(4)})^2} \tag{6.9}$$

Hence, the $\sigma_{u,nm}^{(4)}$ parameter is updated by:

$$\sigma_{u,nm}^{(4)}(t+1) = \sigma_{u,nm}^{(4)}(t) - \varphi\, \Delta\sigma_{u,nm}^{(4)} \tag{6.10}$$

Layer 4- Consequence Layer: In this layer, the local error $e_{n,m}^{(4)}$ is calculated using Equation 4.7 as follows:

$$e_{n,m}^{(4)} = \frac{\partial E}{\partial o_{n,m}^{(4)}} = \frac{\partial E}{\partial y_n^{(5)}} \frac{\partial y_n^{(5)}}{\partial o_{n,m}^{(4)}} = \\ (y_n - y_n^{(5)})\ \frac{\sigma_{u,nm}^{4} \left(c_{u,nm}^{4} \left(\sum_m \sigma_{u,nm}^{4}\ o_{n,m}^{(4)}\right) - \left(\sum_m \sigma_{u,nm}^{4}\ o_{n,m}^{(4)}\ c_{u,nm}^{4}\right)\right)}{(\sum_m \sigma_{u,nm}^{4}\ \ o_{n,m}^{(4)})^2} \tag{6.11}$$

Hence, the adjusted connection weights $W_{u,nm}$ of the links connecting nodes RL_u in Layer 3 to $OL_{n,m}$ in Layer 4 is calculated using Equation 4.6 as follows:

$$\Delta W_{u,nm} = \frac{\partial E}{\partial W_{u,nm}} = \frac{\partial E}{o_{n,m}^{(4)}} \frac{o_{n,m}^{(4)}}{\partial W_{u,nm}} = e_{n,m}^{(4)}\ \frac{o_{n,m}^{(4)}}{\partial W_{u,nm}} \tag{6.12}$$

where

$$\frac{o_{n,m}^{(4)}}{\partial W_{u,nm}} = \begin{cases} o_u^{(3)} & if\ o_{n,m}^{(4)} = \max_{u\epsilon G} \left(o_u^{(3)}\, W_{u,nm}\right) \\ 0 & otherwise \end{cases} \tag{6.13}$$

where G is the set of indices of the nodes RL_u in Layer 3 that are connected to node $OL_{n,m}$ in Layer 4. Hence, the $W_{u,nm}$ parameter is updated by:

$$W_{u,nm}(t+1) = W_{u,nm}(t) + \varphi\, \Delta W_{u,nm} \tag{6.14}$$

Layer 3 - Rule-based Layer: only the local error $e_u^{(3)}$ needs to be computed and propagated using Equation 4.5 as follows:

$$e_u^{(3)} = \frac{\partial \mathrm{E}}{o_u^{(3)}} = \frac{\partial \mathrm{E}}{\partial o_{n,m}^{(4)}} \frac{\partial o_{n,m}^{(4)}}{\partial o_u^{(3)}} = \frac{\partial \mathrm{E}}{\partial o_{n,m}^{(4)}} = e_{n,m}^{(4)} \tag{6.15}$$

Layer 2 - Condition Layer: In this layer, the centres $c_{n,m}^{(2)}$ and widths $\sigma_{n,m}^{(2)}$ of the membership functions of input-label nodes are adjusted using Equation 4.4 as follows:

$$\Delta c_{n,m}^{(2)} = \frac{\partial E}{\partial o_{n,m}^{(2)}} \frac{\partial o_{n,m}^{(2)}}{\partial c_{n,m}^{(2)}} = \frac{\partial E}{\partial o_{n,m}^{(2)}}\, o_{n,m}^{(2)}\, \frac{2\left(o_n^{(1)} - c_{n,m}^{(2)}\right)}{(\sigma_{n,m}^{(2)})^2} \tag{6.16}$$

$$\Delta\sigma^2_{n,m} = \frac{\partial E}{\partial \sigma^{(2)}_{n,m}} = \frac{\partial E}{\partial o^{(2)}_{n,m}} \frac{\partial o^{(2)}_{n,m}}{\partial \sigma^{(2)}_{n,m}} = \frac{\partial E}{\partial o^{(2)}_{n,m}} \; o^{(2)}_{n,m} \frac{2(o^{(1)}_n - c^{(2)}_{n,m})^2}{(\sigma^{(2)}_{n,m})^3} \tag{6.17}$$

where from Equation 4.5:

$$\frac{\partial E}{\partial o^{(2)}_{n,m}} = \begin{cases} \sum_u e^{(3)}_u & if \;\; o^{(2)}_{n,m} = \min_{i \epsilon U} \left(o^{(2)}_i\right) \\ 0 & otherwise \end{cases} \tag{6.18}$$

where U is the set of indices of the nodes in layer 2 that are connected to node u in layer 3. Hence, the $c^{(2)}_{n,m}$ parameter is updated by:

$$c^{(2)}_{n,m}(t+1) = c^{(2)}_{n,m}(t) + \varphi \, \Delta c^{(2)}_{n,m} \tag{6.19}$$

and the $\sigma^2_{n,m}$ parameter is updated by:

$$\sigma^2_{\mathrm{n,m}}(t+1) = \sigma^2_{\mathrm{n,m}}(t) + \varphi \, \Delta\sigma^2_{n,m} \tag{6.20}$$

6.4 Experimental Results and Analysis

In this section, we validate the performance of the FNN-Tool by using a well-known benchmark example (i.e. the Box–Jenkins time series) and show the merits and capabilities of the FNN-Tool as compared to other models. Well-known bench-mark examples are used for the sake of easy comparison with existing models.

6.4.1 Nonlinear System Identification Example

Here, the FNN-Tool is applied to a nonlinear system identification, using the gas furnace data (series J) of Box and Jenkins [16]. The data set used in this experiment was recorded from a combustion process of a methane-air mixture. During the process, the portion of methane was randomly changed, while maintaining a constant gas flow rate. The data set used here, which consists of 296 input-output pairs, is provided in Appendix A. The gas flow into the furnace is the input $x(t)$ and the CO_2 concentration in the outlet gas is the output $y(t)$. The sample interval is 9s. Figures 6.2, 6.3 and 6.4 demonstrate some characteristic functions of the CO_2 time series data. It is noted that there is a finite memory effect indicated by a positive correlation. In addition, the histogram shows significant deviation from the Gaussian behaviour.

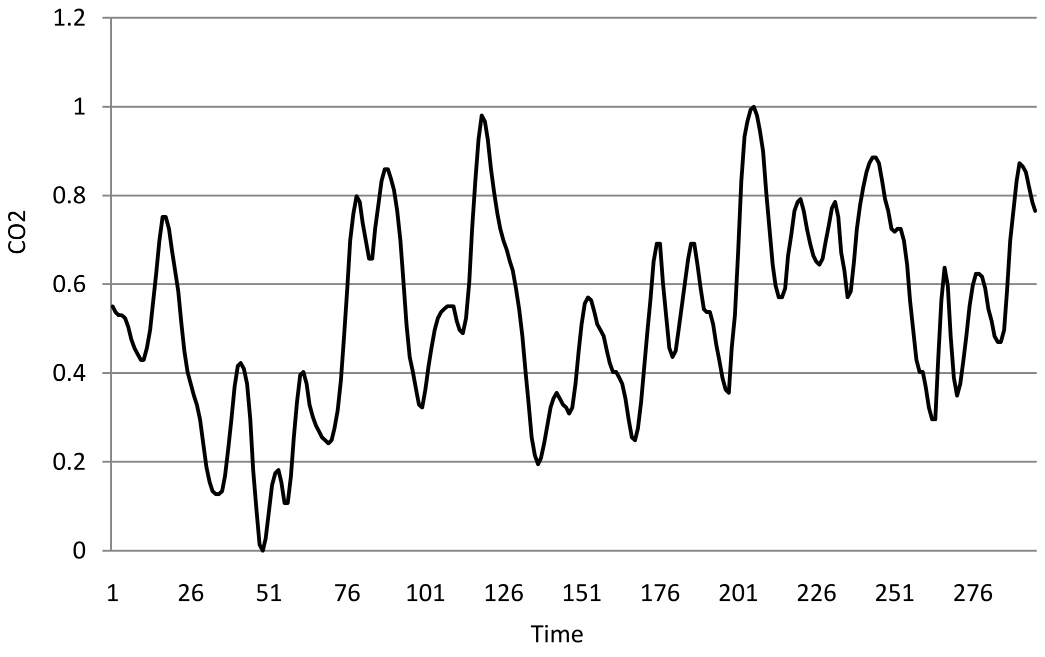


Figure 6.2: The characterization of the Box-Jenkins gas furnace data: original time data.

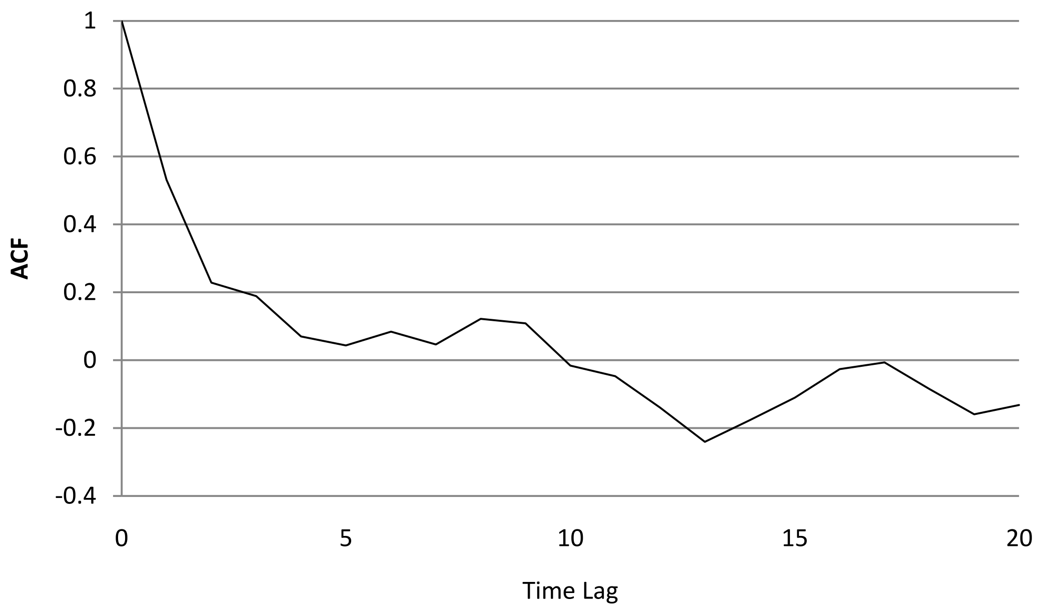


Figure 6.3: The characterization of the Box-Jenkins gas furnace data: auto correlation function.

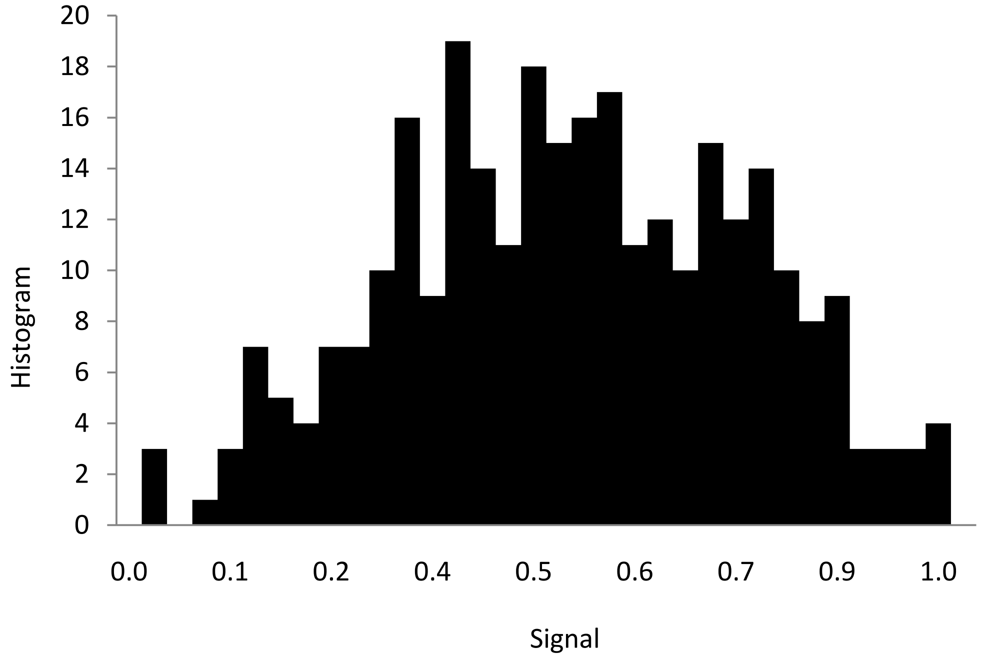


Figure 6.4: The characterization of the Box-Jenkins gas furnace data: Histogram.

6.4.2 Experimental Design

The FNN-Tool is employed to provide identification for the CO_2 concentration $y(t)$. In order to compare the FNN-Tool results with other models, a similar experimental design has been used. It is assumed that the task is to identify the CO_2 produced in a furnace $y(t)$ at time t, given the methane gas portion from four time steps before $x(t-4)$ and the last CO_2 produced in the furnace $y(t-1)$. The data set was converted to $[x(t-4), y(t-1): y(t)]$ pairs which reduced it to 292 input-output pairs.

All data sets used in this experiment were normalized using the *min-max* normalization technique given as follows:

$$X_n = \frac{(X - X_{\min})}{(X_{max} - X_{min})} \qquad 6.21$$

where $X_n$is the normalized value, and X, X_{min}, and X_{max} are an instance of the minimum and the maximum values of the vector to be normalized. The normalized data was then divided such that 70% (204 records) was used for training the FNN-Tool, and the other 30% (88 records) was used for testing the trained FNN-Tool.

This identification problem is then mapped onto the five-layer FNN-Tool with the following configuration as shown in Figure 6.5. The input layer consists of two nodes: $x(t-4)$ and $(t-1)$, whereas, the output layer consists of one node: $y(t)$. The input and the output variables were divided into five linguistic labels (VS, S, M, L, and VL). Thus, the Condition Layer consists of 10 nodes and the Consequence Layer consists of 5 nodes. The initial fuzzy-rule layer consists of all possible combinations of input variables which, in this is case, is 25 nodes. They are fully connected with the Consequence Layer.

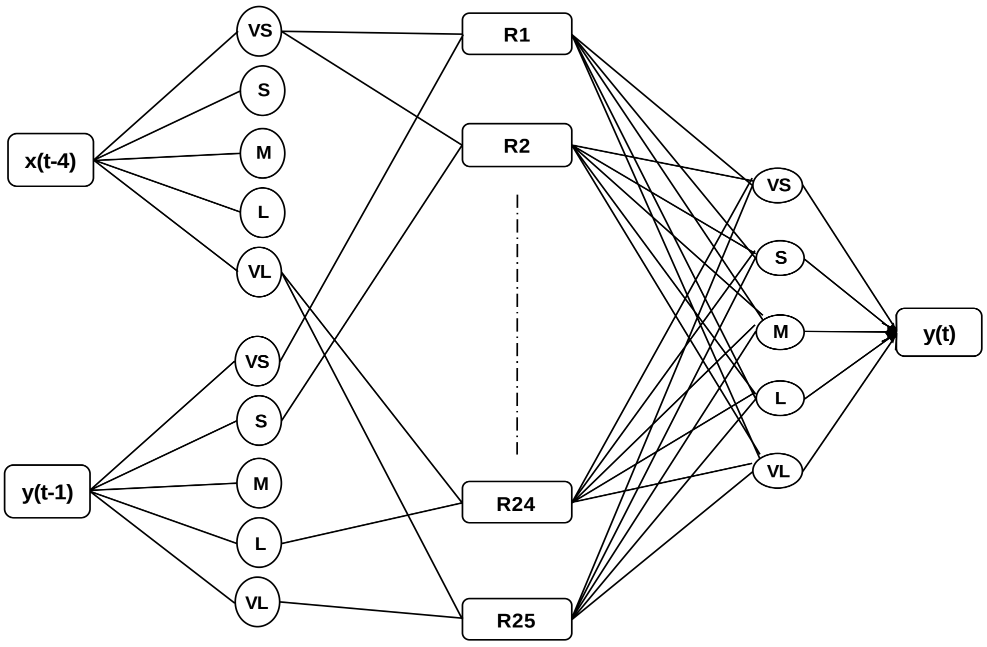


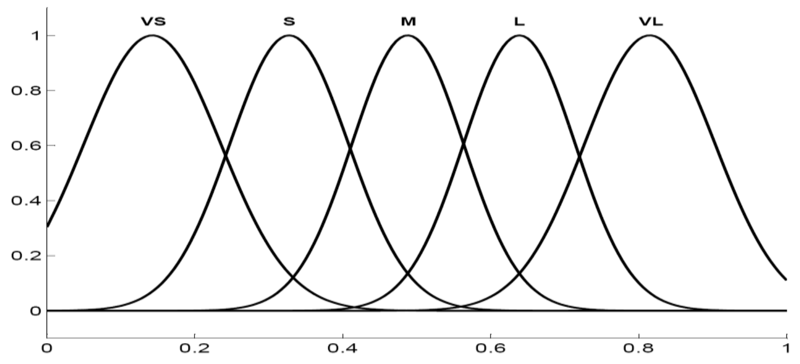
Figure 6.5: The initialized FNN-Tool Structure for the Box-Jenkins gas furnace data

The initial parameters (centre and width) of the membership functions for all inputs and output variables were generated using the self-organizing algorithm described in Section 6.3.1. Figure 6.6 shows these initial membership functions.

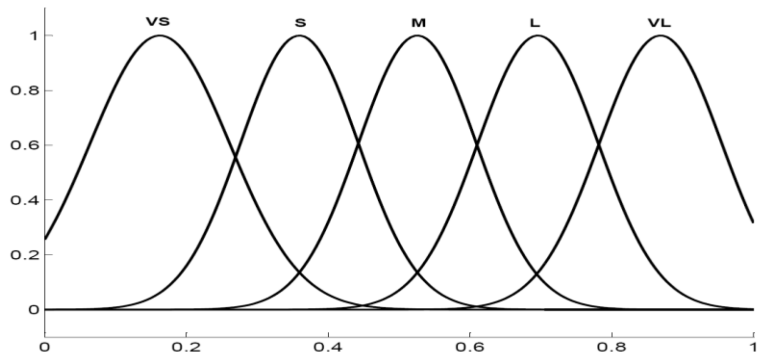
In this experiment we use the Mean Square Error (*MSE*) as a performance index for the FNN-Tool:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \qquad 6.22$$

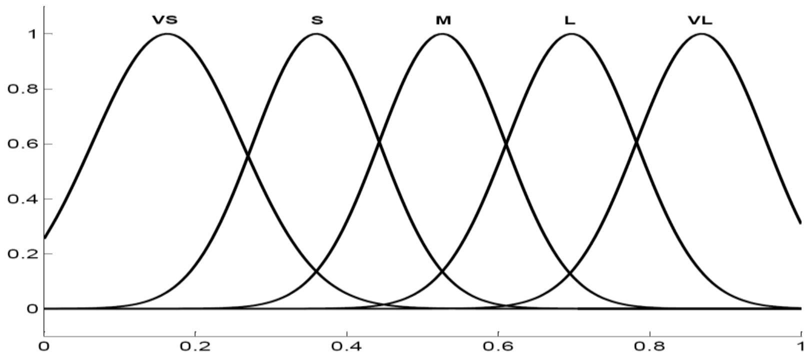
where *N* is the number of data, y_i the actual output, and $\hat{y}_i$ is the model output for *i*th data.



(a) Initial input 1: $x(t-4)$



(b) Initial input 2: $y(t-1)$



(c) Initial output 1: $y(t)$

Figure 6.6: The initial membership functions generated from the first stage of the learning process of FNN-Tool.

6.4.3 GA Operations and Parameters

The GA-based fuzzy rules identification method has been used to generate the fuzzy rules. Firstly, a candidate solution has been encoded into an integer string as a chromosome. The chromosome size has been set as the total number of nodes in the fuzzy-rules layer which is 25. Each gene g_i , where $0 \leq g_{\mathrm{i}} \leq 5$, represents a fuzzy rule. Then an initial population containing *N* chromosomes has been generated randomly. After a series of trial experiments, we found that the optimal learning parameters for this stage were: (*i*) the population size = 90, (*ii*) tournament selection, (*iii*) the elitist generation replacement, (*iv*) standard two-point crossover with probability = 0.7, (*v*) and mutation priority = 0.05. The stopping criterion for a GA run is to achieve the pre-specified error level (e.g. $MSE < 0.0014$).

Figure 6.7 presents an average performance graph for 10 experiments for the selected GA learning parameters. The solid line shows the plot of generation number versus maximum fitness, while the dotted line shows the average fitness of the population. It can be observed from Figure 6.7 that the GA starts with a poor initial population but it improves rapidly. As the GA progresses, the bad solutions are disappeared because their fitness are reduced, so their chance becomes low for the selection process for the next generation. After a few generations a number of good solutions have been found which have been fine-tuned to improve their fitness.

In order to evaluate the impact of different parameters and operations on the GA performance, sensitivity analysis on crossover probability and type, mutation probability, and population size has been carried out. A mutation probability of 0.05 was considered to show how sensitive the GA performance is with different crossover types including one point and two-point and different crossover probabilities ranging from 0.5 to 0.9. All other parameters were kept constant.

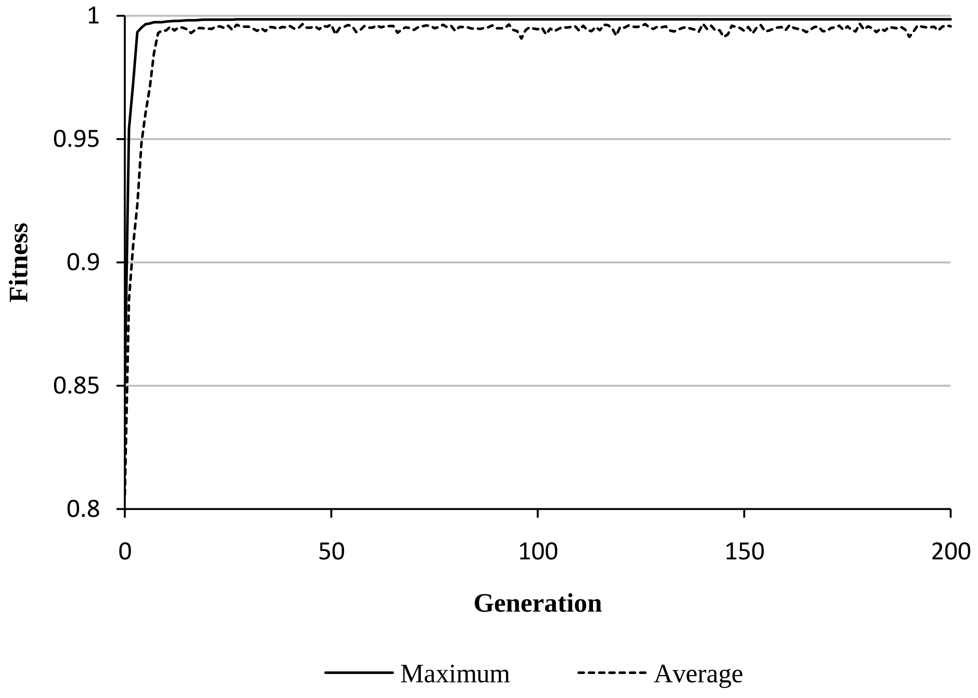


Figure 6.7: GA average performance of 10 runs using a population size of 90, two-point crossovers with probability of 0.7, and a mutation probability of 0.05.

Figure 6.8 illustrates the sensitivity of the GA performance to crossover type and probability. It can be observed from Figure 6.8 that the peak performance of GA was achieved when two-point crossover with a probability of 0.7 was used. The figure also shows that too low crossover probability might lead to a bad performance of GA, especially with one-point crossover.

In the same way, the sensitivity of the performance of GA to mutation probability has been analyzed using a two-point crossover with a probability of 0.7. Figure 6.9 shows how sensitive the GA performance is to different mutation probabilities. It is clear from the figure that the best performance was obtained with mutation probabilities of 0.05 and 0.06. We have considered 0.05 because it is lower. The GA performance are not good if too low (e.g. 0.001 and 0.005) or too high (e.g. 0.2) mutation probabilities are used.

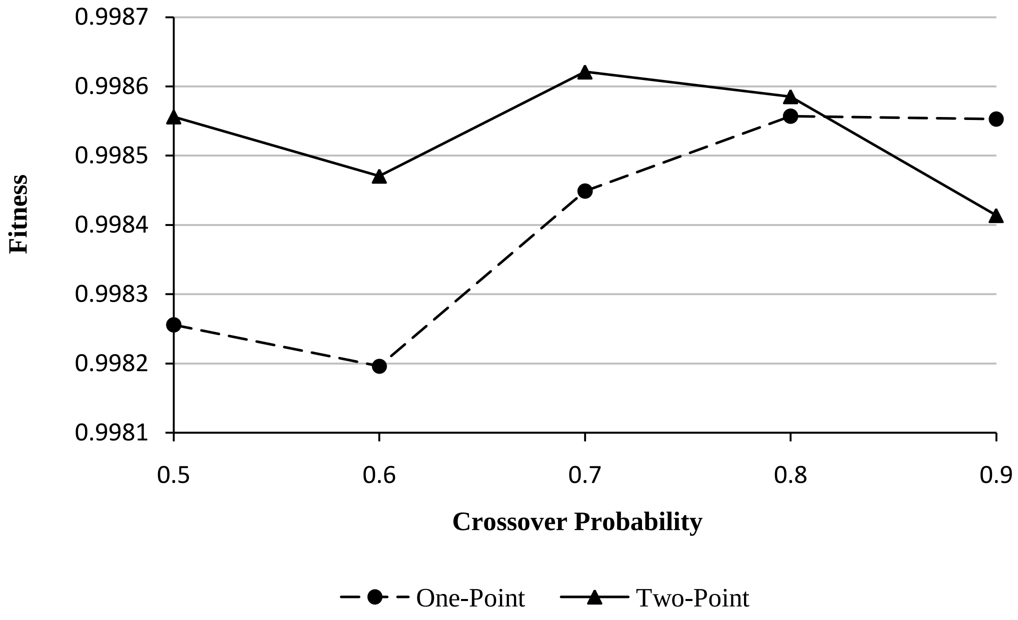


Figure 6.8: Sensitivity to crossover type and probability using mutation probability of 0.05.

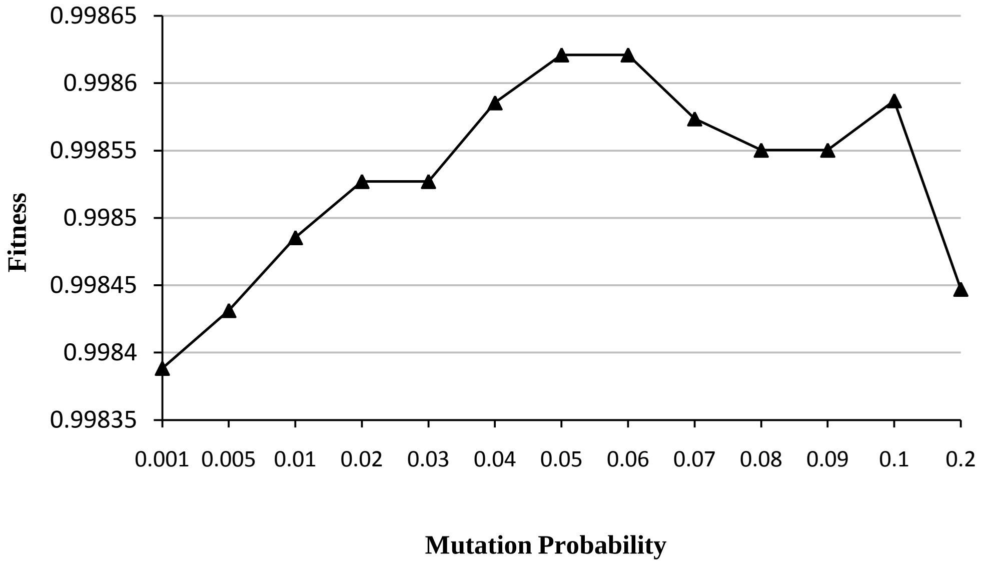


Figure 6.9: Sensitivity to mutation probability using a two-point crossover with a probability of 0.7.

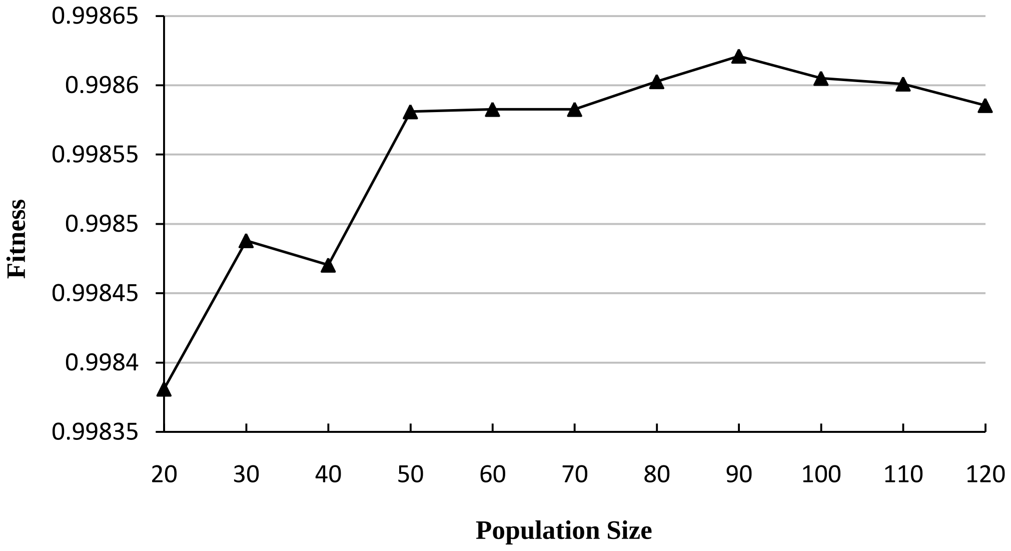


Figure 6.10: Sensitivity to population size using a two-point crossover with probability of 0.7 and a mutation probability of 0.05.

Using a two-point crossover with a probability of 0.7 and a mutation probability of 0.05, the impact of the population size on the GA performance has been analyzed. The sensitivity of the performance of GA to different population sizes ranging from 20 to 120 is illustrated in Figure 6.10. The acceptable performance is obtained with a population greater than 80. The peak performance of GA was achieved with population sizes of 90. The smaller population size decreases the GA performance because the diversity in the population cannot be maintained. In general, the selection of appropriate population size depends upon the user's experience.

6.4.4 Experimental Results

This second stage started with 25 fuzzy rules (i.e. all possible rules), and after completing it, the number of rules was decreased to 18, with a mean square error (*MSE*) = 0.00138. The fuzzy rules generated from this stage are shown in Table 6.1. It should be noted that the weight of all generated rules in this stage were set to 1.

Rule #	IF		THEN	Weight	
	$x(t-4)$	$y(t-1)$	$y(t)$	After Stage2	After Stage3
1	VS	VS	VL	1	0.93
2	VS	S	VS	1	0.94
3	VS	M	L	1	1.00
4	VS	VL	VL	1	0.99
5	S	S	S	1	0.48
6	S	M	L	1	0.17
7	S	L	L	1	0.63
8	S	VL	VL	1	0.31
9*	M	S	S	1	0.00
10	M	M	M	1	1.00
11	M	L	M	1	0.21
12*	L	S	S	1	0.00
13	L	L	L	1	0.79
14	L	VL	VL	1	0.48
15	VL	VS	VS	1	0.68
16*	VL	M	S	1	0.00
17	VL	L	S	1	0.20
18	VL	VL	VS	1	0.20

Table 6.1: Fuzzy rules generated from the second and the third stages of the learning process and their corresponding weights of the Box-Jenkins gas furnace data.

Lastly, after identifying the FNN-Tool structure, the BP algorithm explained in Section 6.3.3 was employed (with learning rate $\varphi = 0.1$) to fine tune the membership parameters and the weights of the fuzzy rules. In this stage two experiments were run. In the first experiment, the BP algorithm was employed only to adjust the memberships parameters (centres, widths) as done in most similar works such as [90] (i.e.

the error for nodes in Layer 4 was computed and propagated without updating to $W_{u,nm}$) In the second experiment the membership parameters and the fuzzy rules weights $W_{u,nm}$ were adjusted. The results of both experiments after 1,000 epochs are illustrated in Figure 6.11. The improvement of the training error rate (*MSE*) in experiment one is denoted by the dotted curve, from 0.00138 to 0.00082, while the solid curve represents the improvement of the error rate (*MSE)* in experiment two, from 0.00138 to 0.00042.

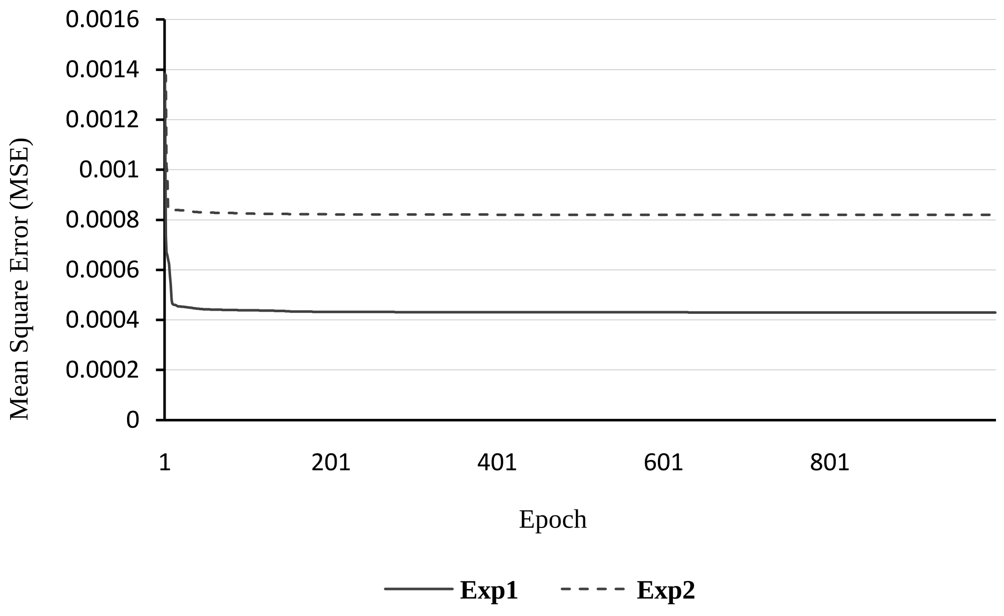
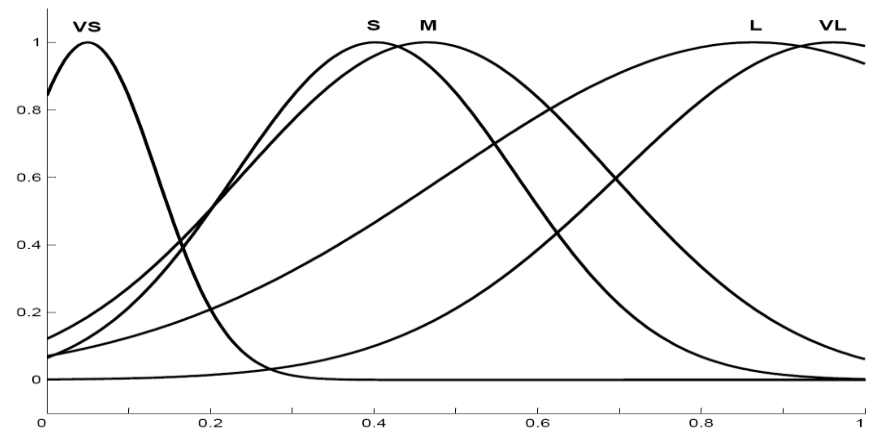
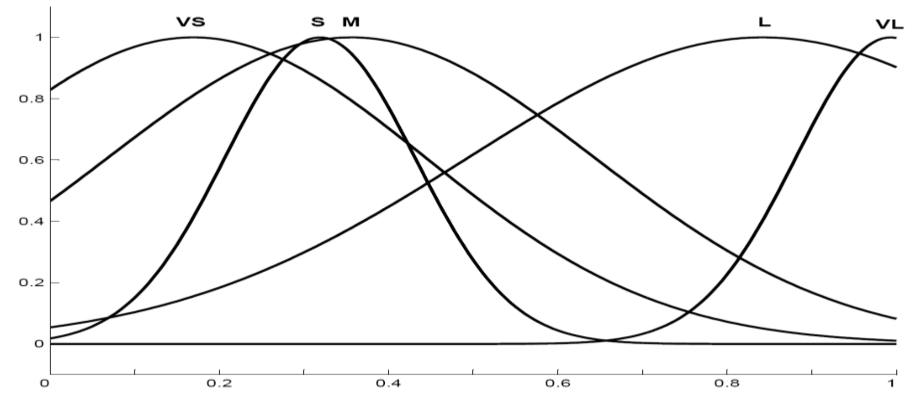


Figure 6.11: The improvement of the FNN-Tool prediction error of the Box-Jenkins gas furnace data after the BP algorithm.

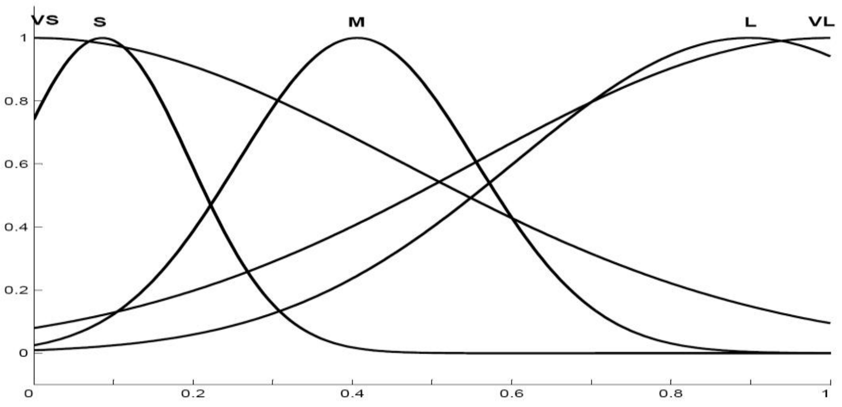
Figures 6.12(a), (b) and (c) show the learned input-output membership functions after this stage. The fuzzy rules and their corresponding weights resulted from this stage are given in Table 6.1. There are three rules with zero weights, outlined by the (*) in the table, which means that those rules can be deleted without affecting the outputs.



(a) Final input 1: $x(t-4)$

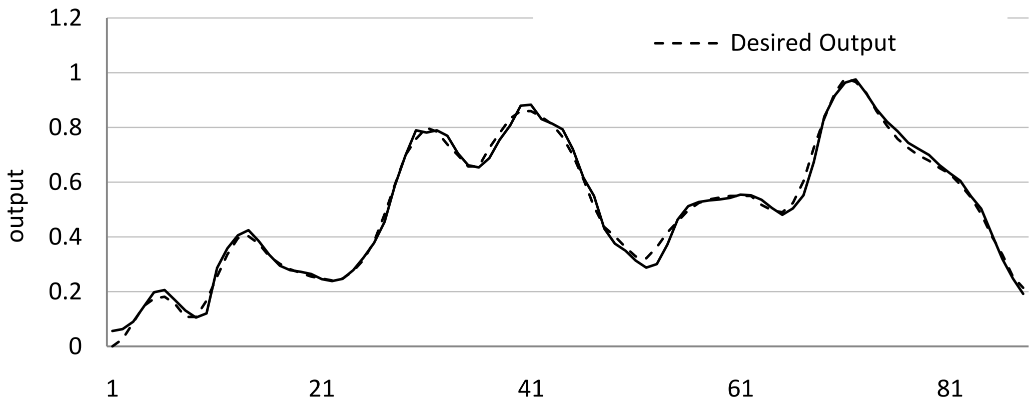


(b) Final input 2: $y(t-1)$

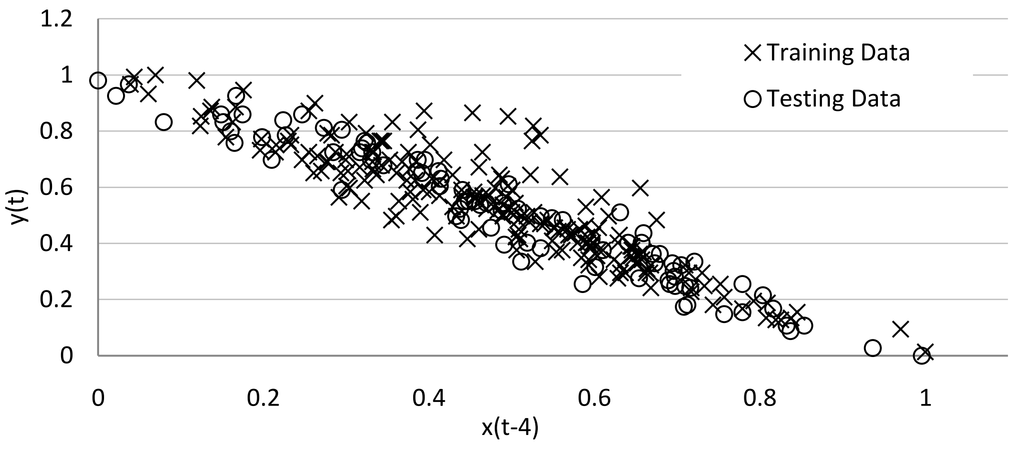


(c) Final output 1: $y(t)$

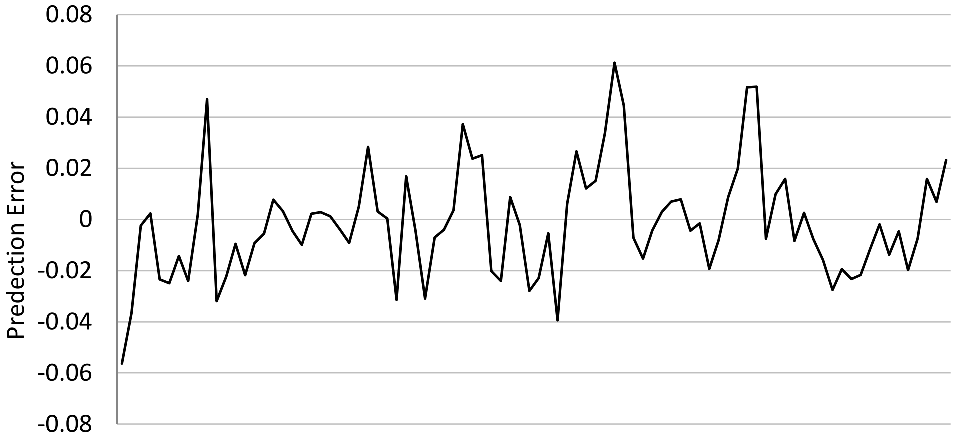
Figure 6.12: The learned input-output membership functions of the Box-Jenkins gas furnace data.



(a)



(b)



(c)

Figure 6.13: The FNN-Tool performance of the Box-Jenkins gas furnace data: (a) desired and predicted output; (b) training and testing distribution; (c) the GAFNN prediction error.

After completing the training process for the FNN-Tool, the testing data set was used to test the trained FNN-Tool. The FNN-Tool successfully identified the desired nonlinear system dynamics with a very low *MSE* (= 0.00045). Figure 6.13 summarizes the final results of this experiment, where the real (desired) and the predicted outputs in response to the testing input data are given in graphical form in Figure 6.13(a), and training and testing data distribution is shown in Figure 6.13(b). Figure 6.13(c) shows the FNN-Tool prediction errors.

6.4.5 Comparison with Other Models

Table 6.2 shows a comparison of the results obtained from our proposed FNN-Tool and that of different models reported in the literature for the Box-Jenkins data prediction problem. The table summarizes the general structure of each model, including the number of inputs and the number of rules. The fuzzy based neural models (such as ANFIS, FuNN, HyFIS, and FNN-Tool) have a much better performance with respect to their counterpart mathematical models (such as ARMA) and fuzzy models (such as Xu, Sugeno and Takagi). This indicates that the fuzzy neural network models are utilizing the advantages of both fuzzy techniques and neural networks in fuzzy neural topology.

It can be observed from the table that the performance of our proposed FNN-Tool is better than the performance of other models, except the HyFIS model. It is worth noting that the HyFIS approach uses all data samples to construct the model (i.e. creating fuzzy membership functions of the input-output variables and generating the fuzzy rules) and 92 data pairs of the same data set were used for testing the model. In our case, the data set was divided into two parts. The first part was used to construct and learn the FNN-Tool model, then the model was tested with the second part which is unseen data.

Model name and reference	Number of inputs	Number of rules	Model error (MSE)
ARMA model **[**16**]**	5	-	0.71
Tong's model **[**160**]**	2	19	0.469
Pedrycz's model **[**129**]**	2	81	0.320
Xu's model **[**181**]**	2	25	0.328
Sugeno's model **[**153**]**	2	6	0.355
Surmann's model **[**154**]**	2	25	0.160
Linear model **[**152**]**	5	-	0.193
Takagi-Sugeno model **[**152**]**	6	2	0.068
Position-gradient model **[**152**]**	3	6	0.190
Lee,s model **[**99**]**	2	25	0.407
Hauptmann's model **[**52**]**	2	10	0.134
Lin's model **[**103**]**	5	4	0.261
Nie's model **[**120**]**	4	45	0.169
Pedrycz's model **[**130**]**	2	25	0.3950
ANFIS model **[**78**]**	2	25	0.00073
FuNN model **[**87**]**	2	7	0.00051
HyFIS model **[**90**]**	2	15	0.00042
FNN-Tool (current model)	*2*	*15*	*0.00045*

Table 6.2: Comparative results for different modelling approaches

Also, the comparative results show that FuNNs is a very promising model, which has an acceptable error level (0.00051) with a much lower number of rule nodes (7) than our FNN-Tool (15). FuNNs use a multilayer perceptron (MLP) network and a modified backpropagation training algorithm. The general FuNN architecture consists of five layers: input, condition, rule, action and output layers. The second and third (condition and rule) layers are fully connected and the sigmoidal logistic activation function is used in each rule node to represent the degree to which input data match the antecedent component of an associated fuzzy rule. That means each rule node in the FuNN is represented by several fuzzy rules, each of them representing a combination of the input condition elements which would activate that node. Therefore, the number of rule nodes in FuNN, which is in Gas-Furnace Time Series example 7, does not represent the total number of fuzzy rules as in our FNN-Tool model. In general, the results of this experiment show that the proposed FNN-Tool is a competitive model when compared with other promising models published in the literature.

6.5 Road Traffic Case Study

In Section 4.4, we created a road traffic case-study for a part of the traffic network in the city of Riyadh in Saudi Arabia, in order to assess the technical feasibility of ITC-DSS. The FNN-Tool was used to predict the Total Travel Time (*TTT*) and the Total Distance Travelled (*TDT*) for the five traffic control actions. (See Chapter 4, Section 4.4 for more details of the case study design). This section presents the learning process and the performance test of the FNN-Tool used in that case study.

The data needed for the training and testing processes of the FNN-Tool in this study have been generated using the traffic macroscopic simulation model METANET as mentioned before. The process of generating traffic data using METANET is explained in the following sub-section.

6.5.1 Traffic Data Generation

Real data is very important for accurately evaluating and testing any new system. However, in some cases, obtaining real data is very difficult. In order to explore the possibility of obtaining real road traffic data for experimentation purposes, we visited and contacted several road traffic control centres and other traffic organizations such as Transportation, Design and Planning of the City of Bradford Metropolitan District Council, The Highways Agency in Manchester, Riyadh Region Traffic Department, Minnesota Traffic Research Laboratory at the University Of Minnesota and the ARC Intelligent Transport Systems Research Laboratory at The University of Queensland. However, the available real traffic data was very expensive to gather, manipulate and process, and also was still not sufficient to cover the entire problem domain. For example, for a road traffic incident, the performance of all possible control actions is required, but such a scenario may have never happened.

The applications of different control actions to real life situations to gather the required data, were not possible for safety reasons. Therefore, we decided to use artificial (simulated) traffic data in this study for training and testing the proposed intelligent traffic control decision support system. However, since the proposed intelligent traffic system employs FNN-Tool, it is easy to modify the fuzzy rule base and the structure of the network when real data become available.

Since the main objective of this ITC-DSS is to help in controlling a traffic network section as a whole without the need to study the individual behaviour of vehicles, we have considered a macroscopic traffic simulation model rather than a microscopic one. The macroscopic traffic simulation model used was METANET, details of which have been provided in Chapter 2 (Section 4). The main reason behind the selection of METANET was not only because a macroscopic approach makes it very suitable for model-based traffic control, but also because it provides a good trade-off between simulation speed and accuracy. Furthermore, the small number of parameters makes it easy to execute and calibrate.

In order to simulate our case study using METANET, the part of the traffic network in the Riyadh city under consideration was represented by a directed graph, where bifurcations, junctions and on/off-ramps were represented by the nodes of the graph and the highway stretches between these locations were represented by the links. Each link is subdivided into segments with typical lengths of 300 to 800 metres. Figure 6.14 shows the network under consideration drawn via METANET. Each link has its geometric characteristics such as the number of lanes, curvature, etc. At the boundaries of the network, origin or destination links were added where traffic enters or leaves, respectively, the part of the traffic network under consideration. Before running METANET, the following information must be declared:

- Simulation control information including: simulation period, simulation time step, incident modelling, etc.
- Traffic data including: traffic demand, traffic density, speed, composition rates, splitting rates, turning rates, etc.
- Control actions including: control start time, end time, where the control action starts and ends, etc.

A simulation run starts with reading the simulation control information and all control actions to be modelled during the simulation time for the initialization stage. The main simulation loop then starts. For each time step, the simulation reads the required traffic data until the simulation loop is terminated. At the end of the simulation, some final information (e.g. control action performance criteria) is produced.

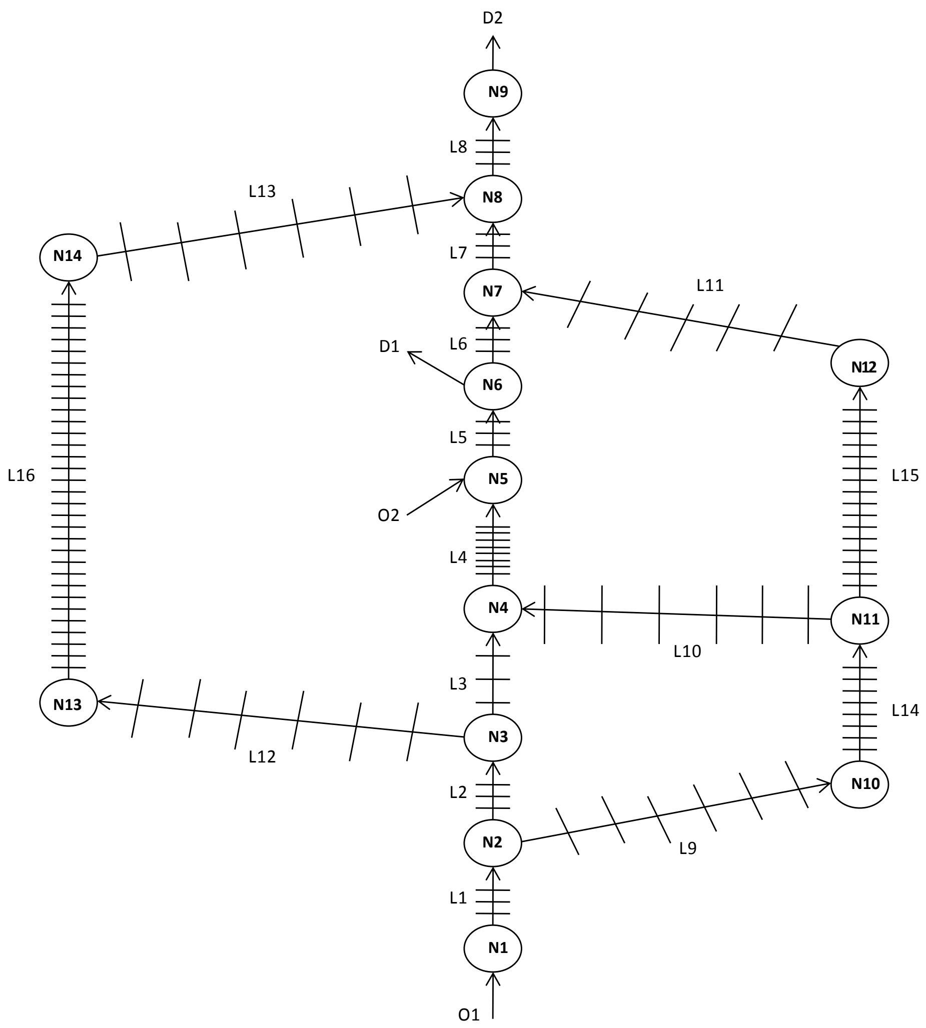


Figure 6.14: The considered traffic network drawn via METANET

For example, in order to generate the required data for the training process of our FNN-Tool for the case study, the simulation has been run many times with different combinations of traffic states, with the five control actions for a prediction time interval set to 120 minutes (i.e. the simulation period). From each run one data pair sample, (ca_i, *TDm, TDn, IS;* TTT_i, TDT_i) has been abstracted. ca_i, *TDm, TDn,* and *IS* are the input variables and they represent the current traffic state and the proposed control action. TTT_i, TDT_i are the output variables and represent the predicted total travel time and predicted total distance travelled after 120m of the application of control action ca_i.

The generated data set consists of 5,000 input-output pairs. Table 6.3 summarizes the statistical indicators for the input-output parameters, including the min, max, mean, standard deviation, and coefficient of variance. It is necessary to ensure that the generated data is sufficient to cover the entire problem domain, including maximum and minimum values for each variable, as well as a good distribution of values within this range. Using the values in Table 6.3, the normalization of data samples was performed using Equation 6.21. The normalized data samples are then divided such that 70% was used for training the FNN-Tool and 30% unseen data was used for testing the trained FNN-Tool.

	TDm	*TDn*	*IS*	*TTT*	*TDT*
MIN	500.0	5.0	2000.0	807.7	76685.6
MAX	6000.0	150.0	6000.0	58606.2	352915.9
Mean	3311.6	79.4	4094.0	23244.6	23244.6
Std Dev	1608.2	43.1	1620.9	16549.2	55863.3
CV (%)	48.6	54.3	39.6	71.2	240.3

Table 6.3: Maximum, minimum, and average value of data set for the considered traffic case study.

6.5.2 Learning FNN-Tool

The membership functions of both input and output variables were initially determined as follows. The input variables *TDm*, *TDn* and *IS* were divided into three labels (low, medium, and high), whereas the control action ca_i was directly represented by five non-fuzzy indices (C1, C2, C3, C4, and C5). On the other hand, the output variables TTT_i and TDT_i were divided into five labels (very low, low, medium, high, and very high). Consequently, the condition layer consisted of 14 nodes and the consequence layer consisted of 10 nodes. The fuzzy-rule layer consisted of all possible combinations of input variables which, in this case, was 135 nodes. They are fully connected with the Consequence Layer. Figure 6.15 shows the overall structure of the FNN-Tool after this stage. The initial parameters (centre and width) of the membership functions for all inputs and outputs variables are summarized in Table 6.4 in columns 3 and 4.

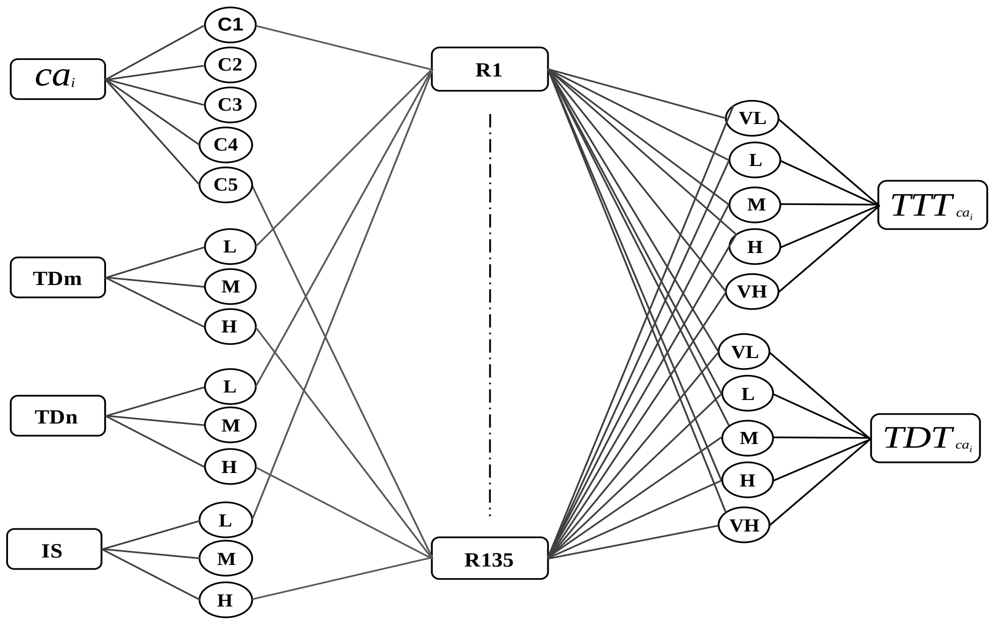


Figure 6.15: the initialized FNN-Tool Structure for the traffic case study.

I/O variables	Label	Initial Parameters		Final Parameters	
		Centre	Width	Centre	Width
ca_i	C1	0	0.105	0	0.105
	C2	0.25	0.105	0.25	0.105
	C3	0.5	0.105	0.5	0.105
	C4	0.75	0.105	0.75	0.105
	C5	1	0.105	1	0.105
TDm	Low	0.16	0.17	0.29	0.48
	Medium	0.50	0.17	0.51	0.41
	High	0.85	0.17	0.88	0.28
TDn	Low	0.15	0.17	0.16	0.19
	Medium	0.50	0.17	0.52	0.17
	High	0.86	0.18	0.79	0.22
IS	Low	0.2	0.21	0.14	0.40
	Medium	0.5	0.21	0.53	0.41
	High	0.9	0.21	0.98	0.32
TTT	Very Low	0.0592	0.09395	0.01	0.1
	Low	0.2471	0.09395	0.03	0.115
	Medium	0.4732	0.0986	0.12	0.12
	High	0.6704	0.0876	0.71	0.215
	Very High	0.8456	0.0876	0.89	0.185
TDT	Very Low	0.2136	0.1234	0.13	0.085
	Low	0.4604	0.0871	0.14	0.035
	Medium	0.6346	0.0654	0.8	0.11
	High	0.7654	0.0654	0.85	0.115
	Very High	0.9094	0.072	0.95	0.095

Table 6.4: Initial and final parameters of the MFs for all variables of the traffic case study.

We started the second stage of the learning process with 135 fuzzy rules, and after completing this stage, 126 relevant fuzzy rules were identified with a mean square error (*MSE*) of 0.013. After a series of experiments the following learning parameters were identified as appropriate: (*i*) the population size = 200, (*ii*) tournament selection, (*iii*) the elitist generation replacement, (*iv*) standard two-point crossover with probability = 0.8, (*v*) and mutation probability = 0.04. The stopping criterion was 1,500 generations. Figure 6.16 shows the FNN-Tool prediction error of the traffic case study after the second stage of the learning process. The curve represents an average performance of 10 experiments.

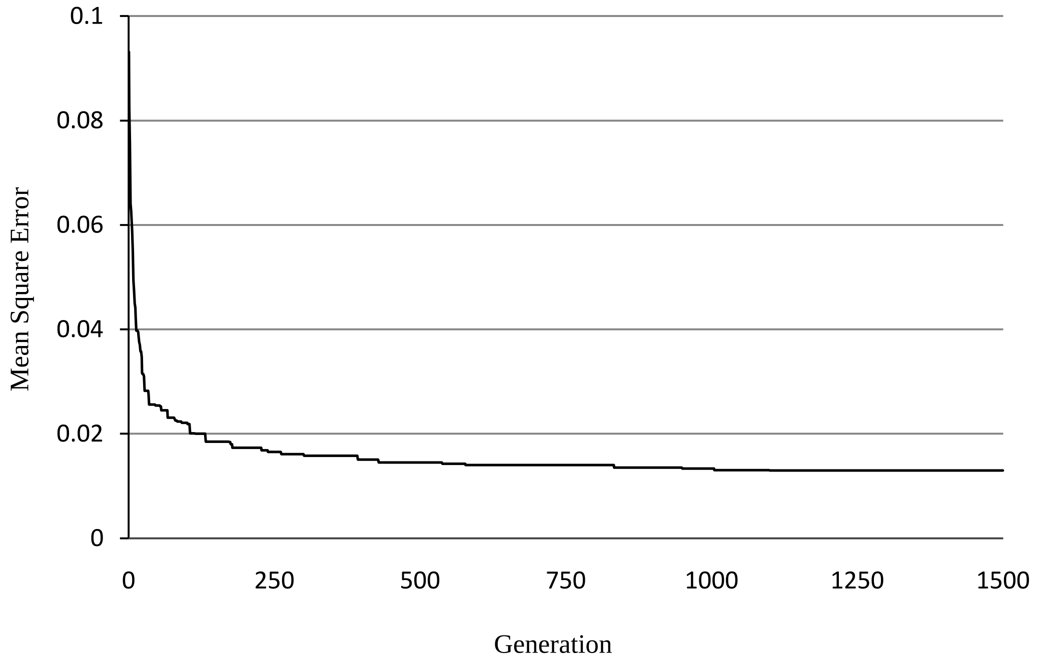


Figure 6.16: The FNN-Tool prediction error of the traffic case study after the second stages of the learning process.

Due to the large number of experiments and the long training time required, a modified master-slave parallel genetic algorithm (used in [139]) was used with a network of 35 PCs to speed up this stage of training. For example, the completion of one generation using only one PC takes approximately 450 seconds, which is decreased sharply to approximately 19 seconds with 35 PCs.

The BP algorithm (with a learning rate φ of 0.1) was then used to fine-tune the membership parameters and the weights of the identified fuzzy rules. The error rate *MSE* was improved to 0.0024 and the number of fuzzy rules decreased to 112 rules. The improvement of *MSE* during this stage is illustrated in Figure 6.17. The final parameters (centre and width) of the membership functions for all input and output variables are summarized in Table 6.4 in columns 5 and 6.

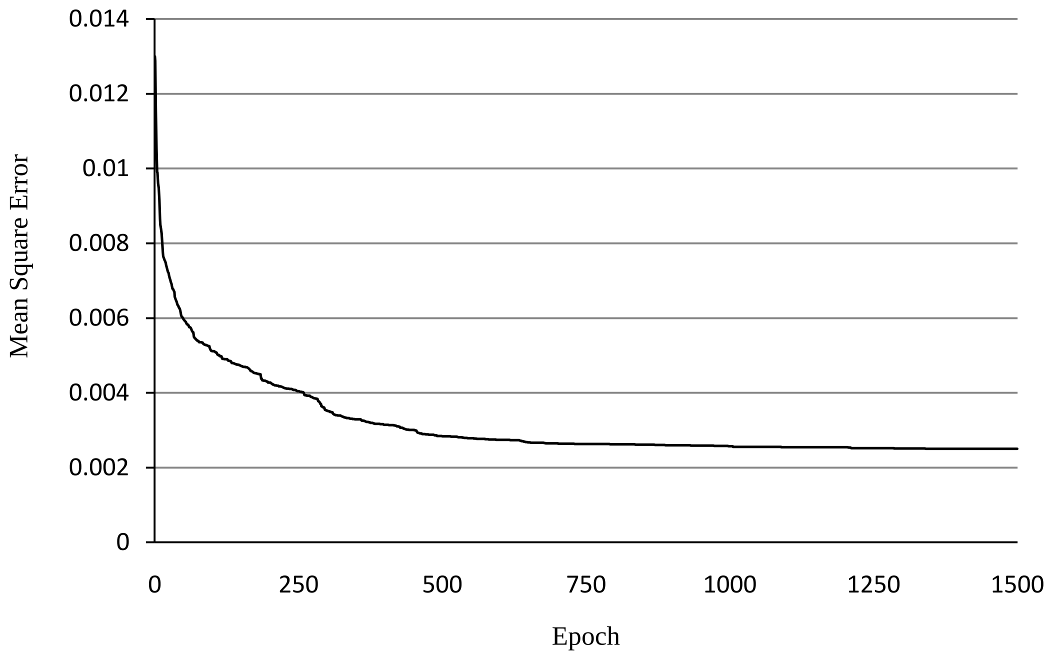
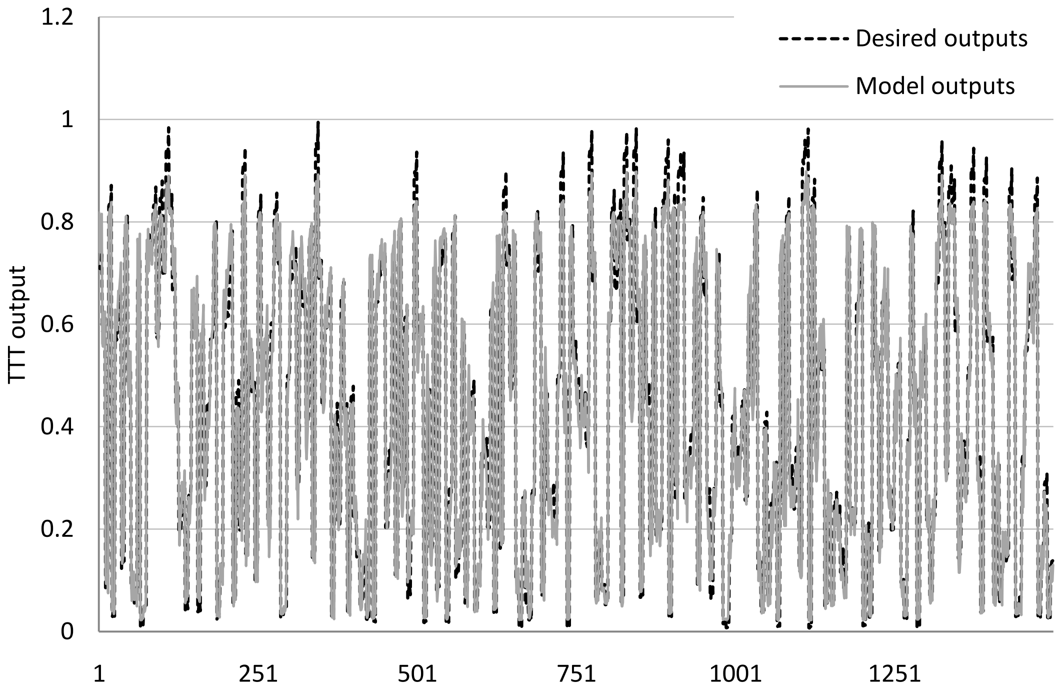
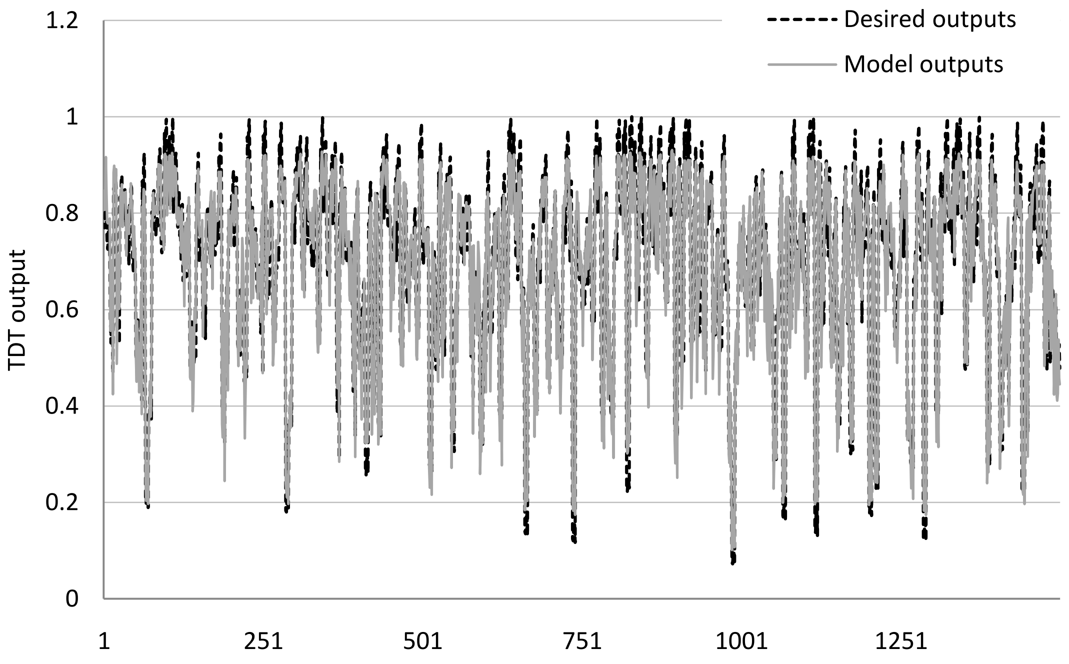


Figure 6.17: The improvement of the error rate after the BP algorithm.

Finally, the trained FNN-Tool was employed to predict the total travel time and the total distance travelled for the five control actions using the testing data. The general testing results are given in Figures 6.18 and 6.19. The real and the modelled outputs in response to the testing input data are illustrated in Figure 6.18 (a) and (b), while the FNN-Tool prediction errors are illustrated in Figures 6.19 (a) and (d). We can see very clearly from these figures that the FNN-Tool can predict the total travel time and the total distance travelled to the desired level of accuracy.

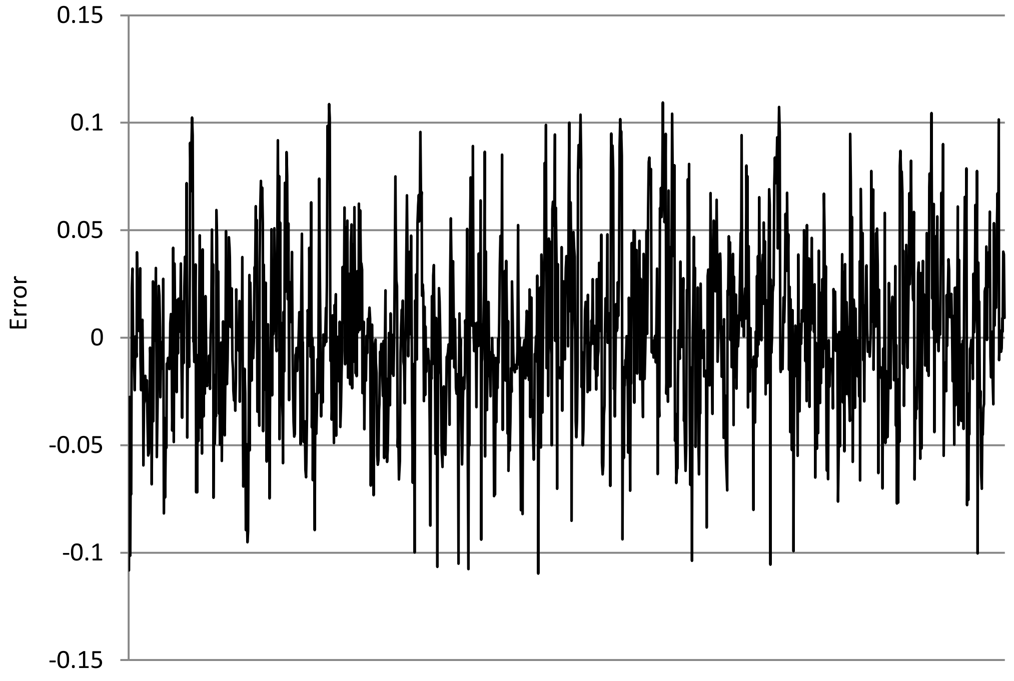


(a)

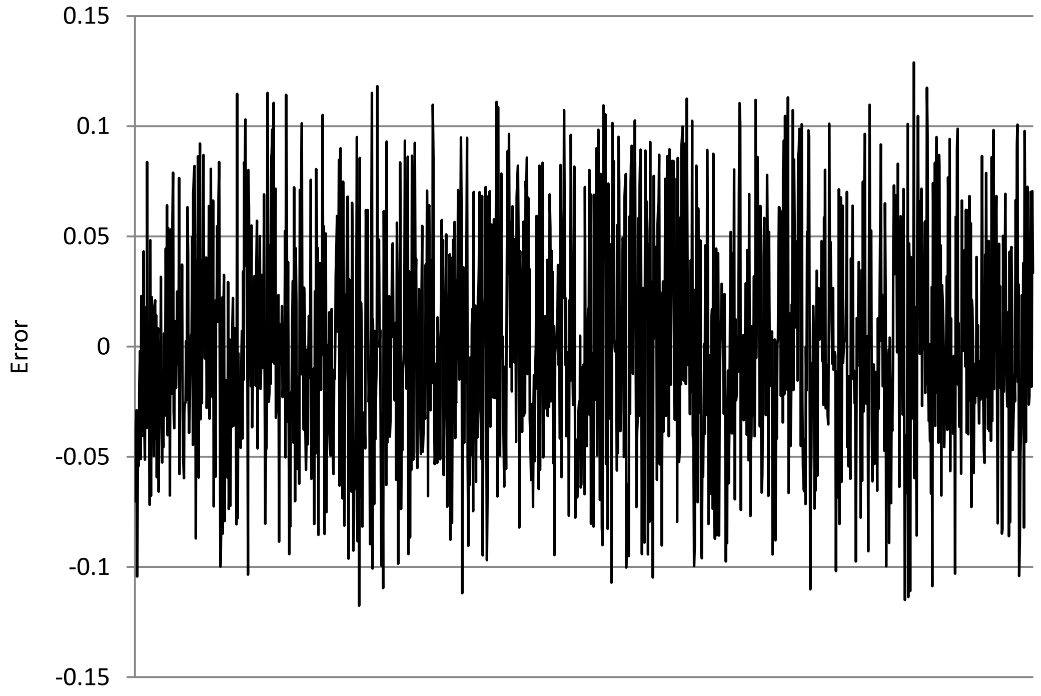


(b)

Figure 6.18: (a) Desired and predicted output *TTT*; (b) Desired and predicted output *TDT*.



(a)



(b)

Figure 6.19: (a) The FNN-Tool prediction error of *TTT*; (d) the FNN-Tool prediction error of *TDT*.

6.5.3 Result Validation

In order to illustrate how accurate the predicted data set models the desired output data set, we used the coefficient of determination (R^2) as a performance index of the FNN-Tool with $0 \leq R^2 \leq 1$. When R^2 equals zero, the desired and predicted outputs are totally uncorrelated. In contrast, when R^2 equals 1, they are exactly the same. R^2 is defined as:

$$R^2 = \frac{\left[\sum_{i=1}^{N}(y_i - \bar{y})(\hat{y}_i - \bar{\hat{y}})\right]^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2 \ \sum_{i=1}^{N}(\hat{y}_i - \bar{\hat{y}})^2} \tag{6.23}$$

where *N* is the number of data, y_i is the *i*th actual output, $\hat{y}_i$ is the *i*th model or predicted output, $\bar{y}$ is the mean of actual output y_i, and $\bar{\hat{y}}$ is the mean of model output $\hat{y}_i$.

The results of the FNN-Tool performance prediction for the total travel time and the total distance travelled were recorded for each control action separately. The model outputs of the FNN-Tool was plotted against the actual outputs to give an indication of the FNN-Tool's ability to predict the total travel time and the total distance travelled while applying the traffic control actions, and the R^2 values were calculated accordingly.Table 6.5 summarizes the final results of this part of the experiment.

Table 6.5 shows that the R^2 values calculated for *TTT* prediction are higher than the values calculated for the *TDT* prediction. This indicates that the level of accuracy of the prediction of *TDT* is lower than the level of accuracy of the prediction of *TTT*. Also, we can see that the FNN-Tool is able to predict the performance of the control action ca_5 more accurately than other control actions. In general, it can be observed that R^2 values obtained from the FNN-Tool are very promising with $R^2 \geq 0.93$ for all control actions, which means that the FNN-Tool is able to accurately predict the total travel time and the total distance travelled for all control actions.

Traffic Control Action	R^2	
	TTT	*TDT*
ca_1	0.97	0.93
ca_2	0.98	0.95
ca_3	0.98	0.95
ca_4	0.97	0.94
ca_5	0.98	0.96

Table 6.5: R^2 of the TTT and TDT for the five control actions

6.6 Summary

This chapter has described an effective learning approach for the FNN-Tool. The proposed learning approach consists of three stages: the first stage is initializing the membership functions of both input and output variables by determining their centres and widths using a self-organizing algorithm; the GA-based method presented in the previous chapter is performed in the second stage to identify the fuzzy rules; in the last stage, the derived structure and parameters are fine-tuned by using the back-propagation learning algorithm.

A well-known benchmark example was used to test the performance of the proposed learning approach. Moreover, the prediction capability of the FNN-Tool trained by the proposed learning approach was assessed for forecasting the performance of traffic control actions on the current traffic state. Experimental results have demonstrated the ability of the proposed learning approach to identify all relevant fuzzy rules from the training data. Comparative analysis has shown that the proposed learning approach has a higher degree of prediction capability than other models.

The main features of the proposed FNN-Tool performs the learning process of identifying the fuzzy rules and the learning process of adjusting rules weights in separate stages to ensure that only the relevant rules are trained. The main advantages of the FNN-Tool and the three stage-based learning approach proposed in this chapter are: (1) the FNN-Tool is a general framework which combines the advantages of two intelligent techniques, namely fuzzy logic and neural networks; (2) the initial set of parameters and the structure of the FNN-Tool have been derived from the numerical information; (3) it is easy to modify the fuzzy rule base and the structure of the FNN-Tool when new data (or linguistic information) becomes available; (4) the FNN-Tool model is applicable for solving different types of problems where prediction is essential.

In the next chapter, the ITC-DSS presented in the previous chapters will be extended to obtain a scalable system using a multi-agent approach.

Chapter 7

A Multi-Agent Approach for Intelligent Traffic Control Systems

7.1 Chapter Overview

In this chapter we propose a multi-agent based approach for road traffic control systems. The system proposed in this chapter is a major extension of ITC-DSS presented in the previous chapters. ITC-DSS was successfully tested for a small-sized network and for a limited number of traffic situations and control actions. However, when ITC-DSS was used for a large-sized network, it did not scale up well. The reason behind that is that a large network, characterizing the current traffic state, requires a large number of traffic variables, as well as the number of possible traffic control actions that can be applied to control the current state can be large. Using ITC-DSS with such large numbers of inputs is not efficient because the training process of the FNN-Tool is overloaded. Therefore, in this chapter we have developed a multi-agent based approach where a large network is divided into a number of sub-networks, each of which has its own ITC-DSS and its own agent. The coordination between agents is achieved through a high level agent called coordinator. The coordinator receives proposed local control actions from the agent of the sub-network with an incident, resolves conflicts between other affected agents, and sends the globally acceptable solution back to that agent.

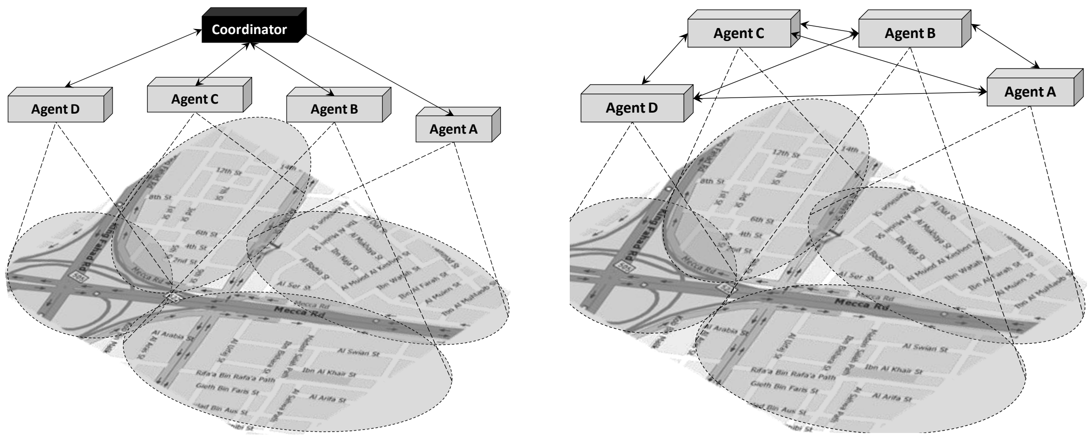
The chapter is structured as follows. Initially, it discusses the need for global network-oriented traffic control. Subsequently, the proposed multi-agent approach is introduced. Finally, the application of the proposed system within a case study is presented in order to demonstrate the capabilities of the proposed system.

7.2 Global Network-Oriented Traffic Control

The traffic control actions influencing a traffic situation in one area of a road network can also influence the traffic situation in neighbouring areas. Therefore, as the spatial interrelations between traffic situations at different locations in the network get stronger, consequently the interrelations between the traffic control actions at different locations also gets stronger. These interrelations may differ depending on the situation (depending also upon network topology, traffic demand, etc.) and the control actions may be cooperative or counteractive. Coordinative control strategies are required in these cases, to make sure that all available control actions serve the same objective. For example, solving local traffic congestion only, can have a consequence that the vehicles run faster into another (downstream) congestion, where still the same number of vehicles have to pass the downstream bottleneck (with a given capacity). In such a case, the average travel time on the network level will still be the same or worse.

In large traffic networks such as metropolitan areas, several agencies share the administration of the transport infrastructure, through a distributed network of traffic operation centres responsible for the management and control of their facilities. Even if the main purpose of these agencies is, in general, the efficient management of the urban network, different agencies have different policies that may generate conflicting operations. Furthermore, the spatial and administrative organization of such agencies often results in a localized distribution of data and information, and in the presence of multiple decision-making entities that pursue different goals and adopt different criteria to achieve these goals. Therefore, the presence of different modes of transportation and the different demand and performance characteristics of interacting subsystems, such as freeways and surface streets, require an intelligent multi-agent control system. Several reports, for example [163] and [48], address the need of inter-agency cooperation for a more efficient resolution of the conflicts that may arise.

Multiagent systems have been applied successfully to a variety of road traffic management problems such as traffic simulations and prediction problems[34], [112] and traffic control problems [29], [30] (See Chapter 3). Initially, most distributed traffic problem-solving systems were based on a distinguished agent, which achieved the coordination of the activities of its acquaintances in a centralized fashion. More recently, the focus has shifted to more autonomous agents that coordinate in a decentralized fashion [62]. Centralized and decentralized multiagent traffic management systems differ significantly in the way that these traffic agents are coordinated (see Figure. 7.1). In a centralized multiagent system, all traffic agents are connected with a high level agent called a coordinator agent. The coordinator agent receives local control proposals from the traffic control agents, resolves conflicts between them, and sends the resulting globally consistent local plans back to the traffic agents. An example of a centralized multiagent system for traffic management is InTRYS[27]. In a decentralized multiagent system, coordination between agents is achieved through communication that takes place on the same hierarchical level. An example of a centralized multiagent system for traffic management is TRYSA2[124].



a) Centralized **b) Decentralized**

Figure 7.1: Centralized and decentralized coordination in multi-agent systems.

7.3 The Proposed Multi-agent System

In this section we describe our proposed multi-agent system. The architecture of the proposed multi-agent system is a centralized coordination, where traffic agents are connected with a high level agent called a coordinator agent. The coordinator agent receives local control proposals from the traffic control agents, resolves conflicts between them, and sends the resulting globally consistent local signal plans back to the traffic agents.

7.3.1 Structure

Consider a traffic network consisting of several highway (motorway/freeway) links. Traffic enters the network via the origin links (e.g. on-ramps or highway links coming from outside the network), and leaves the network via destination links (e.g. off-ramps or highway links going out of the network). The given traffic network is divided into overlapping regions, called sub-networks, and each sub-network is supervised and controlled by an agent. Each agent has three traffic subsystems (see Figure 7.2).

1) Monitoring and detection subsystem: This subsystem determines the current traffic state in the sub-network and detects any traffic problem in the sub-network. It provides the operator with a diagnosis of the problem, together with an explanation for such a diagnosis. Examples of such a subsystem has been reported in [180], [100], and [159].
2) Traffic control subsystem: This subsystem assists the operator to predict the local performance of the proposed control actions using ITC-DSS.
3) Traffic devices control subsystem: This subsystem monitors and controls the available traffic devices such as Variable Message Signs (VMS), traffic lights, ramp meters, etc.

Our research focuses on the traffic control subsystem (ITC-DSS) and how this subsystem can react with other agents to find the most suitable global traffic conditions. The description of the monitoring and detection subsystem and the traffic devices control subsystem is out of the scope of this thesis.

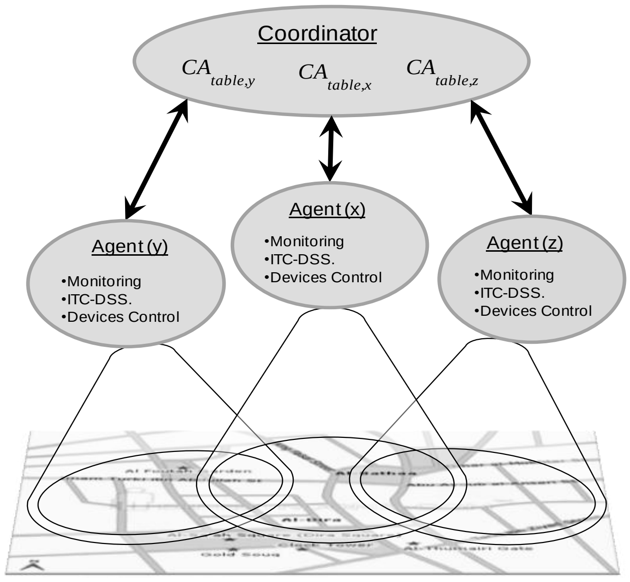


Figure 7.2: The overall structure of the proposed multi-agent system.

The coordinator uses a control actions data-table (CA_{table}) for each of the sub-networks. This control action table is constructed with all possible traffic control actions that can be applied on a sub-network. As mentioned before, a traffic control action can be one control action such as lane closure, ramp metering, variable message signs etc., or a combination of several control actions. The control action data-table is generated for a given sub-network off-line using the available road control facilities, traffic operator's experience, and historical traffic data. This also takes into consideration the interrelations between the traffic control actions at different locations in the network. The overall structure of the proposed multi-agent system is depicted in Figure 7.2.

The structure of CA_{table} is illustrated in Table 7.1. Considering sub-network Z, each record in CA_{table} is characterized by the following:

- Traffic control action (ca_i): name (or description) of the traffic control action that can be applied on the sub-network Z.

- Affected sub-networks (sn_j): all sub-networks that might be influenced by the traffic control action ca_i. This field is characterized by the following:
  - Agent-ID (g_j): the identity (or name) of the agent that controls the sub-network (sn_j).
  - Influence Rates (Y_j^i and R_j^i), where Y_j^i represents percentage change (positive or negative) that may happen in the traffic flows from sub-network Z to the affected sub-network (sn_j) (i.e. sn_j traffic demand) due to the application of the traffic control action ca_i, and R_j^i representing the outflow restrictions for the affected sub-network (sn_j) due to the application of the traffic control action ca_i. The value of R_j^i is given as a percentage and it means that the maximum outflow capacity will be reduced by R_j^i during the application of ca_i. The values of Y_j^i and R_j^i can be estimated using the historical traffic data or alternatively using a traffic simulation program.

Traffic Control Action	Affected sub-networks						
	sn_1 (g_1)		sn_2 (g_2)		. . .	sn_j (g_j)	
ca_1	Y_1^1	R_1^1	Y_2^1	R_2^1	. . .	Y_j^1	R_j^1
ca_2	Y_1^2	R_1^2	Y_2^2	R_2^2	. . .	Y_j^2	R_j^2
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
ca_i	Y_1^i	R_1^i	Y_2^i	Y_2^i	. . .	Y_j^i	R_j^i

Table 7.1: The structure of the proposed control actions data table CA_{table}

7.3.2 Operation

Once the control actions data-tables have been constructed, they are used by the coordinator to identify the best global control actions as follows. Suppose there are 4 traffic agents (*A*, *B*, *C*, and *D*), which control 4 sub-networks (sn_1, sn_2, sn_3, sn_4), respectively. When a traffic problem is detected by the monitoring sub-system controlled by agent *A*, it runs its ITC-DSS to come up with a ranked list of best local control actions (*S*) for the coordinator (as explained in Chapter 4). Each control action of *S* has a fitness value calculated by agent *A* (i.e. local aggregated performance P^i). Let $S = \{ca_1, ca_2, ca_3\}$. Next, all agents that will be affected by any control action of the proposed *S*, are determined by the coordinator ing CA_{table} . Let Agent *B* be affected by ca_1 and ca_2 , Agent *D* be affected by ca_1 and ca_3 , and Agent *C* be not affected (i.e. its traffic state will not be affected by any control action of *S*). In this case, the coordinator will send ca_1 and ca_2 with their associated influence rates (Y_B^1 and Y_B^2) to Agent *B* to calculate their fitness. Similarly, ca_1 and ca_3 , with their associated influence rates (Y_D^1 and Y_D^3), will be sent to Agent *D* to calculate their fitness.

The affected agents (*B* and *D*) will calculate the fitness of the proposed control actions using the influence rates, as we will see in the next section, then will return the results to the coordinator. The global performance of each control action of *S* is predicted by the coordinator using the fitness of the control actions received from agent *A* and the affected agents. The process of calculating the global performance of the control actions is explained in Section 7.4.4. Finally, the proposed control actions of agent *A* (*S*) will be re-ranked by the coordinator based on their global performance and returned to agent *A*. In some cases, agent *A* may need to use a traffic simulation program to effectively compare the best two (or more) control actions before applying them. If, for example, ca_1 has been selected by agent *A* as a best global control action, the coordinator is responsible for informing agents *B* and *D* to guarantee that all applied control actions at that time serve the same objective. Thus, agents *B* and *D* can apply their selected local control actions (if any) simultaneously. A simple flowchart of this process is illustrated in Figure 7.3.

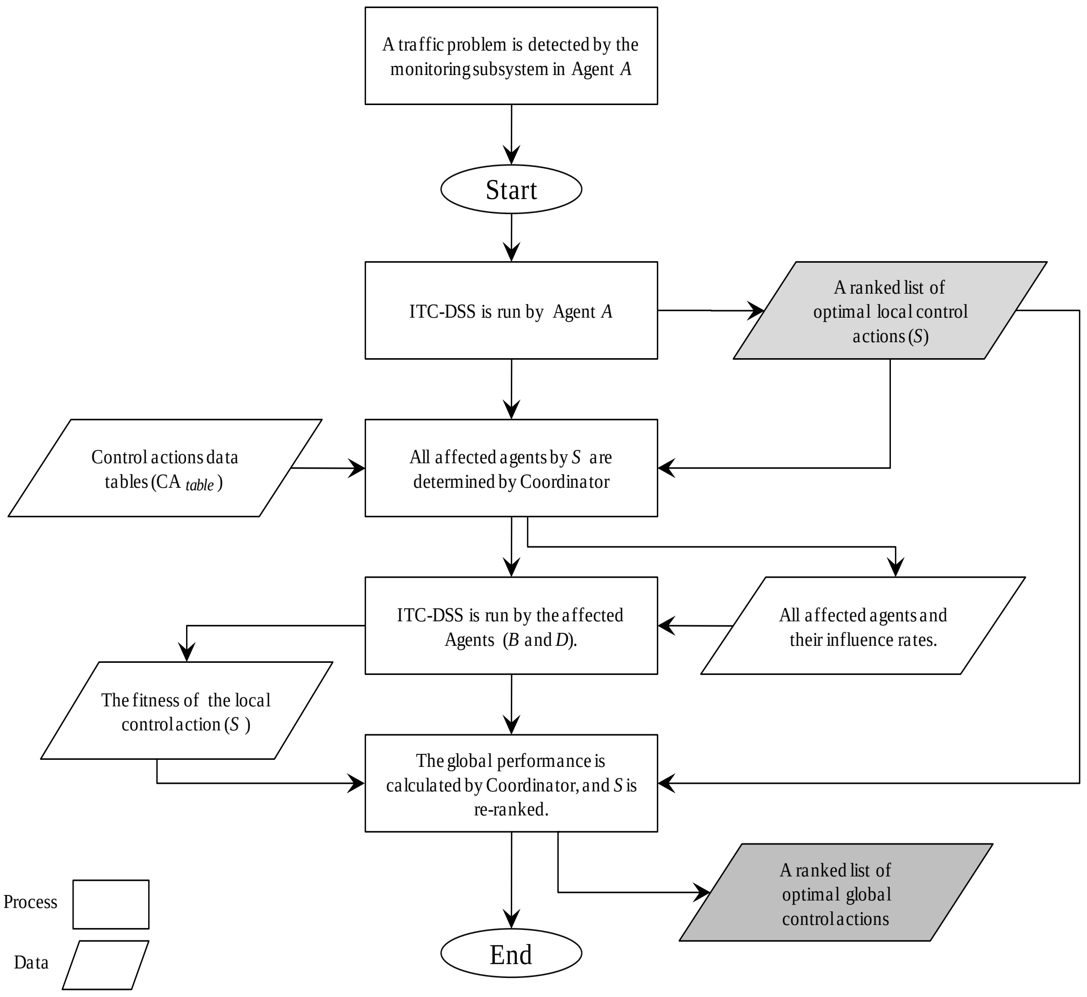

A traffic problem is detected by the monitoring subsystem in Agent *A*
Start
ITC-DSS is run by Agent *A*
A ranked list of optimal local control actions (*S*)
Control actions data tables (CA $_{table}$)
All affected agents by *S* are determined by Coordinator
ITC-DSS is run by the affected Agents (*B* and *D*).
All affected agents and their influence rates.
The fitness of the local control action (*S*)
The global performance is calculated by Coordinator, and *S* is re-ranked.
A ranked list of optimal global control actions
End
Process
Data

Figure 7.3: Flowchart of the process of the proposed multi-agent system.

7.3.3 Calculation of Control Action Fitness

Once the affected agents receive the proposed local control actions (ca_i) with their associated influence rates (Y_j^i and R_j^i) from the coordinator, they run their ITC-DSS subsystems to calculate their fitness for the proposed control actions. According to our example, *A* is the agent which detected the problem and *B* is the agent which is affected by the control action ca_1 and ca_2 . To calculate the fitness of ca_1 , agent *B* runs his pre-trained ITC-DSS. In this case, the input of ITC-DSS will be the current traffic state of sub-network (sn_2) (including any current traffic accidents, the predicted traffic demand, and the outflow restrictions due to the application of ca_1) and only the internal control actions that mainly influence the traffic flows within its network (such as shoulder lane opening or variable speed limits). The main reason behind agent *B* using only the internal control actions is to ensure that the nominated control actions will not have the negative knock on effect of creating a new problem for one or more of its neighbours.

The predicted traffic demand of the sub-network associated with agent *B* (i.e. sn_2) coming from the sub-network associated with agent *A* (i.e. sn_1) due to the application of ca_1 , can be calculated by using the influence rate (Y_B^1) as follows:

$$P\_Dem_B^1 = C\_Dem_B + \left(C\_Dem_B * {}^{Y_B^1}\!/_{100} \right) \tag{7.1}$$

where $P\_Dem_B^1$ and C_Dem_B denote the predicted and the current traffic demand of sub-network sn_2 (associated with agent *B*) coming from sub-network sn_1 (associated with agent *A*) respectively.

Finally, the best internal control action will be selected by agent *B* and its aggregated performance will be sent as agent *B*'s fitness (F_B^1) of ca_1 to the coordinator. The fitness of a control action (F_j^i) is in the range of [0,1]. When F_j^i equals zero, the control action ca_i is totally unsuitable for agent g_j (i.e. for sub-network sn_j). In contrast, when F_j^i equals 1, the control action ca_i is totally suitable.

7.3.4 Global Performance of Control Action

The process of calculating the global performance of a control action is performed by the coordinator. All affected agents will calculate the fitness of the proposed control actions received from the coordinator according to their traffic states, and then return the results to the coordinator. The global performance p_g^i for each proposed local control action ca_i is now determined as:

$$p_g^i = \frac{p_l^i + \sum_{j=1}^{N}(F_j^i \quad \omega_j \quad \mu_j^i)}{1 + \sum_{j=1}^{N}(\omega_j \quad \mu_j^i)} \tag{7.2}$$

where p_l^i is the aggregated performance of the control action ca_i which is calculated by agent *A* (i.e. the local fitness of ca_i of agent *A*); *N* is the number of affected agents; F_j^i is the fitness of ca_i which is provided by the affected agent g_j (according to our example $g_j \in \{B,D\}$); the weights $\omega_j > 0$ represent the relative importance of agent g_j. The weights (ω_j) are not necessarily fixed, but can be changed on-line by the coordinator, depending on the current traffic management policies and other considerations; and μ_j^i is a measure that shows how much impact the traffic control action ca_i is having on the affected sub-network associated with agent g_j. When μ_j^i is high, ca_i has a high impact on the traffic state of the sub-network associated with agentg_j. When μ_j^i is low, ca_i has a low impact on the traffic state of the sub-network associated with agent g_j. μ_j^i is calculated by the coordinator using influence rates (Y_j and R_j) from CA_{table} as follows :

$$\mu_j^i = \begin{cases} \left({Y_j^i}/{100} * \frac{P_{Dem}{}_j^i}{Max_{OF_j}} \right) + {R_j^i}/{100} & if\ Y_j^i > 0 \\ {R_j^i}/{100} & otherwise \end{cases} \tag{7.3}$$

where $P\_Dem_j^i$ is the predicted traffic inflow into the affected sub-network associated with agent g_j coming from the sub-network associated with agent *A* due to the application of ca_i (i.e. traffic demand of agent *j*). $P_{Dem}{}_j^i$ can be calculated by using Equation 7.1. Max_{OF_j} is the maximum possible traffic inflow into the affected sub-network associated with agent g_j coming from the sub-network associated with agent *A* (e.g. the maximum capacity of the links between two sub-networks).

When two or more different traffic problems are detected by different agents, the coordinator is responsible for ranking those agents based on their importance at that time, with the most important one being considered first. The other agents with lower priority will consider their traffic problems as a part of their current traffic states when they use their ITC-DSS to calculate the fitness of the local control actions proposed by that agent (the most important agent).

7.4 Case Studies and Experimental Results

7.4.1 Riyadh Traffic Case Study

In order to test the technical feasibility of the proposed multi-agent system, a traffic case-study was created for a part of the traffic network in the city of Riyadh in Saudi Arabia (see Figure 7.4). The selected network, which is an extension of our case study discussed in Chapter 4, consists of three parallel highways (the Olaya highway, the King Fahad highway, the Takhassusi highway) connected via several on- and off-ramps. In this case study, we only consider traffic going from the south to the north (i.e. towards the city centre). Traffic enters the network from five origins (*O1*, *O2*, *O3*, *O4*, and *O5*) and leaves the network through six destinations (*D1*, *D2*, *D3*, *D4*, *D5*, and *D6*). We have divided the network into three sub-networks King Fahad, Olaya, and Takhassusi (see Figure 7.4), controlled and managed by three agents *A*, *B*, and *C*, respectively. It is worth noting that the King Fahad sub-network alone represents the part that was considered in the case study presented in Chapter 4.

The sub-networks have been simulated separately using METANET (as shown in Chapter 6) for several traffic states and different control actions with the following parameters:

- Incidents vary in severity from 20% to 80% of reduction in link capacity;
- Simulated time period two hours (e.g. from 9:00 am to 11:00 am);
- The traffic entering the King Fahad sub-network is divided as follows:

20% goes to destination *D1* (97% uses the King Fahad highway, and 3% uses the Olaya highway), 55% goes to destination *D2* (90% uses the King Fahad highway, 6% uses the Takhassusi highway, and 4% uses the Olaya highway), while the rest goes to destinations *D3* (5%), *D4* (10%), *D5* (5%), and *D6* (5%);

- The traffic entering the Olaya sub-network is divided as follows: 20% goes to destination *D3*, 60% goes to destination *D4,* and the rest goes to destinations *D1* (10%) and *D2* (10%);

- The traffic entering the Takhassusi sub-network is divided as follows: 70% goes to destination *D6*, 20% goes to destination *D5*, and the rest goes to destinations *D2* (10%);

- The traffic state has been represented by: average traffic demand (*TDm*), average traffic density (*TDn*), incident severity (IS), and outflow restrictions (*OFR*). The generated data has been used to train the ITC-DSS and to create a $\mathrm{CA}_{\mathrm{table}}$ for each sub-network.

Note that the results presented in this section are not intended to verify the ability of ITC-DSS to correctly predict the best local control actions, because this has already been done in Chapters 4 and 6. The aim is to demonstrate the technical feasibility of the proposed multi-agent approach, and to show how the traffic agents react with the coordinator in order to effectively identify the best global control action from a set of control actions.

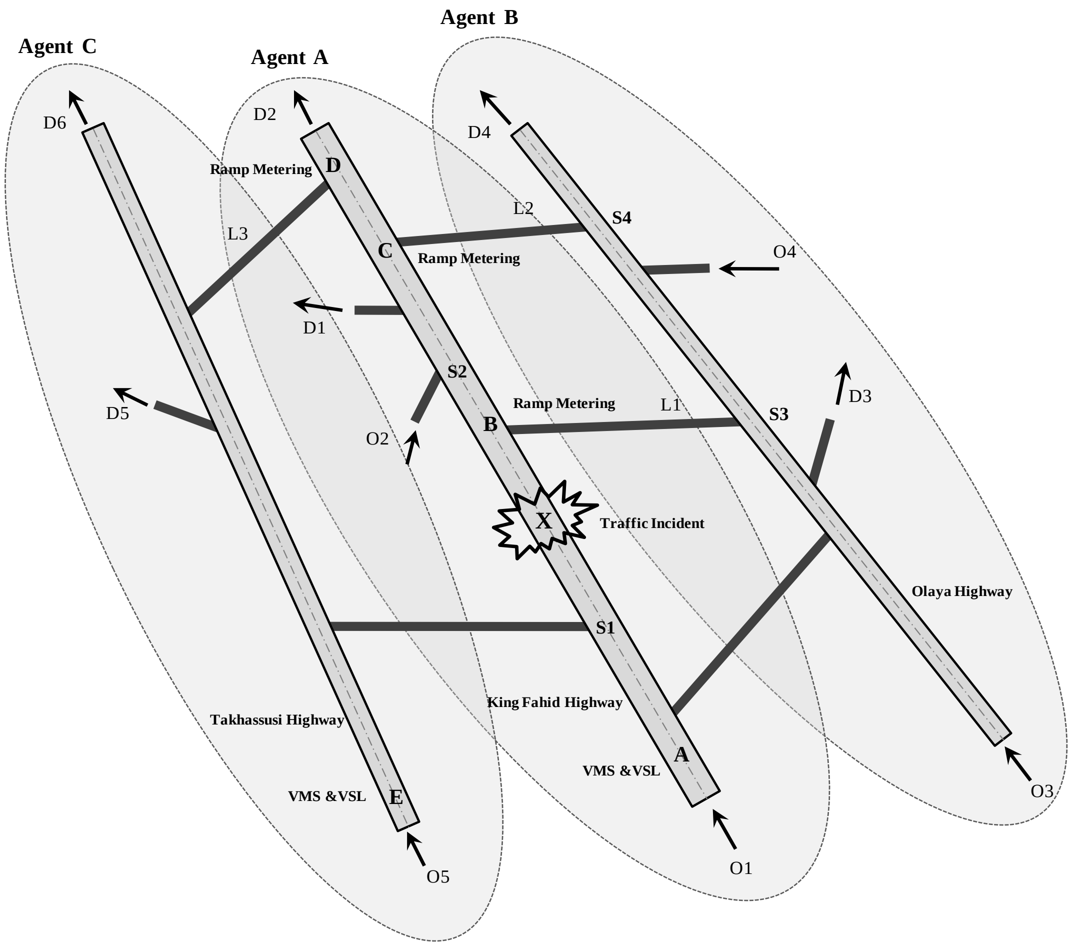


Figure 7.4: The Riyadh traffic network considered in the case study.

7.4.2 Application of the Proposed Multi-agent Approach

In order to make a comparison with the results obtained in the case study presented in Chapter 4, i.e., to illustrate the difference between best local control actions and best global control actions, we are considering here the following similar control actions.

- ca_1: Using VMS at point *A* to direct traffic that goes to *D2* to use the Olaya highway.

- ca_2: Using VMS at point *A* to direct traffic that goes to *D1* to use the Olaya highway.
- ca_3: Using VMS at point *A* to direct traffic that goes to *D1* and traffic that goes to *D2* to use the Olaya highway.
- ca_4: Using VMS at point *A* to direct traffic that goes to *D2* to use the Takhassusi highway.
- ca_5: Using VMS at point *A* to direct traffic that goes to *D1* to use the Olaya highway and on Ramp Metering at point *B*.

The considered current traffic states of the three sub-networks are summarized in Table 7.2. For comparison purposes, we consider the same traffic state of the King Fahad sub-network presented in Chapter 4 and we have assumed a heavy traffic state in the Olaya sub-network, and a smooth traffic state in the Takhassusi sub-network.

Traffic Variables	King Fahad sub-network	Olaya sub-network	Takhassusi sub-network
TDm	6800	7500	4000
TDn	32	24	9
IS	75%	0%	0%
OFR	0%	0%	0%

Table 7.2: The considered traffic states of the three sub-networks.

The current and the maximum possible traffic demands (C_Dem_j and Max_OF_j) of Olaya and Takhassusi sub-networks coming from King Fahad sub-network are shown in Table 7.3.

Traffic Variables	Olaya sub-network	Takhassusi sub-network
C_Dem_j	943.4	809.84
Max_OF_j	4000	4000

Table 7.3: The current and the maximum possible traffic demands (C_Dem_j and Max_OF_j) of Olaya and Takhassusi sub-networks coming from King Fahad sub-network.

Creating CA_{table} for the King Fahd sub-network

As we mentioned before, a CA_{table} can be created using historical traffic data or a traffic simulation model. In this experiment we have used the METANET program. Table 7.4 shows the CA_{table} for the King Fahad sub-network with the five control actions (ca_1 , ca_2 , ca_3 , ca_4 and ca_5). Due to the fact that the Olaya sub-network is connected with the King Fahad sub-network via two outflow links *L1* and *L2* (see Figure 7.4), the restriction of the outflow capacity (R_j^i) field is represented by two columns (columns 3 & 4). For example, as can be seen from the table, the traffic control ca_1 affects the traffic demand on the Olaya sub-network negatively by 207.6% and the traffic demand on the Takhassusi sub-network positively by 24.2%. That is, ca_1 increases the traffic inflow into the Olaya sub-network from the

King Fahad sub-network by 207.6%, and decreases the traffic inflow into the Takhassusi sub-network from the King Fahad sub-network by 24%. This is because 55% of the traffic that enters the King Fahad sub-network and needs to go to *D2* is directed by ca_1^A.

Control Actions	Affected sub-networks				
	Olaya sub-network (Agent *B*)			Takhassusi sub-network (Agent *C*)	
	Y_B^i	R_B^i		Y_C^i	R_C^i
		L1	*L2*		
ca_1	+207.6%	0%	0%	-24.2%	0%
ca_2	+76.3%	0%	0%	0%	0%
ca_3	+283.9%	0%	0%	-24.2%	0%
ca_4	-8.7%	0%	0%	+225.1%	0%
ca_5	+76.3%	30%	0%	0%	0%

Table 7.4: CA_{table} for the King Fahd sub-network with the five control actions.

Table 7.4 also shows that only ca_5 reduces the maximum outflow capacity *L1* of the Olaya sub-network by 30%, while there is no negative effect on the outflow of the Takhassusi sub-network from any control action. It is worth noting that the restriction of outflow capacity of a link does not have that much of a negative impact if the traffic outflow in that link is low. For example, if the maximum capacity of a link is 4,000 vehicles/hour, 30% restriction mainly affects the sub-network outflow when the traffic outflow of that link is more than 2,800 vehicles/hour.

Calculation of Control Action Fitness

Agents B and C (the affected agents) calculate the local fitness of ca_1 , ca_2 , ca_3 , ca_4 and ca_5 using their ITC-DSS, as explained earlier. Table 7.5 summarizes the values of w_{C_d} , C_d^{min} and C_d^{max} used by agents *B* and *C* to calculate the aggregated performance (fitness) of each control action. The table shows that w_{TDT} is assigned 0 by agent *B*, indicating that *TTT* is the only considered performance criterion at that time. As mentioned before, the values of w_{C_d} , C_d^{min} and C_d^{max} are not fixed. They are assigned by the operator based on the traffic state or other policies.

Traffic sub-networks	*TTT*			*TDT*		
	w_{TTT}	TTT_{min}	TTT_{max}	w_{TDT}	TDT_{min}	TDT_{max}
King Fahad sub-network	1.5	3000	10000	0.5	80000	250000
Olaya sub-network	1.0	3300	8800	0	150000	300000
Takhassusi sub-network	1.5	1800	7000	0.5	180000	250000

Table 7.5: the values of w_{C_d} , C_d^{min} and C_d^{max} used by agents *B* and *C* to calculate the aggregated performance of each control action

Table 7.6 and Table 7.7 summarize the best performance of the five control actions ca_1 , ca_2 , ca_3 , ca_4 and ca_5 predicted by agents *B* and *C* respectively after considering the traffic demands and the outflow restrictions imposed by agent *A*, and their internal control actions. In this part of the experiment we have considered two internal control actions for Agent *B*: 1) doing nothing; 2) shoulder lane opening on section (*S3* - *S4*); and two internal control actions for agent *C*: 1) doing nothing; 2) reduce the speed limit from 90 k/h to 60 k/h using VSL at point *E* (see Figure 7.4). For each control action, E_{TTT} and E_{TDT} have been calculated using Equation 4.2 and summarized in columns 3 and 5. The aggregated performances (fitness) of the control actions, which will be returned to the coordinator, are calculated using Equation 4.1 and listed in column 6 of each table.

Control Actions	TTT	E_{TTT}	TDT	E_{TDT}	Aggregated performance (Fitness)
ca_1	7500.35	0.24	237209.50	0.42	0.24
ca_2	4771.25	0.73	235074.10	0.43	0.73
ca_3	8796.80	0.00	232126.80	0.45	0.00
ca_4	3373.71	0.99	247009.50	0.35	0.99
ca_5	5935.66	0.52	222331.50	0.52	0.52

Table7.6: The predicted performance of the five control actions for agent *B*.

It is observed from Table 7.6 that ca_4 is recommended by agent *B* (with fitness 0.99) because the application of ca_4 does not increase the traffic inflow to Olaya sub-network, while ca_3 is completely rejected (with fitness 0.0) because it increases the Olaya sub-network inflow which affects the sub-network traffic state very negatively. ca_5 has a moderately negative impact on the inflow and the outflow of the Olaya sub-network and has 0.52 fitness. On the other hand, Table 7.7 shows that agent *C* strongly recommends all control actions except ca_4 , which increases the traffic inflow to the Takhassusi sub-network. However, ca_4 is still accepted by agent *C* (with fitness 0.62), because the traffic flow in the Takhassusi sub-network at that time is smooth.

Control Actions	TTT	E_{TTT}	TDT	E_{TDT}	Aggregated performance (Fitness)
ca_1^A	1792.28	1.00	180820.40	0.99	1.00
ca_2^A	1871.61	0.99	187451.30	0.89	0.97
ca_3^A	1792.28	1.00	180820.40	0.99	1.00
ca_4^A	2753.42	0.82	248387.90	0.02	0.62
ca_5^A	1871.61	0.99	187451.30	0.89	0.97

Table 7.7: The predicted performance of the five control actions for agent *C*.

Calculation of Control Action Global Performance

Table 7.8 summarizes the final results of the proposed multi-agent system for calculating the global performance for the five control actions. The table shows that the Olaya sub-network is assigned a larger weight, $\omega = 1$, than the Takhassusi sub-network ($\omega = 0.5$), indicating the high importance of the Olaya sub-network at that time. The μ values of each sub-network, which indicates how the sub-network was affected by the control actions, are computed for each control action using Equation 7.3 and summarized in columns 4 and 6. The local and the global performance of the control actions are given in the second and the last columns of the table, respectively.

It is observed from Table 7.8 that the best global control action is ca_4 which can reduce the congestion in the King Fahad sub-network and also improve the overall traffic state in the network. Although ca_3 has the best local performance to solving the traffic congestion in the King Fahad sub-network, it is not the best global control action. That is simply because ca_3 passes a large number of vehicles from the King Fahad sub-network to the Olaya Sub-network at a time when the Olaya sub-network is suffering from bad traffic, which will not improve the overall traffic state in the network overall.

Control Action (Ranked)	Local Aggregated Performance (King Fahad sub-network)	Olaya sub-network		Takhassusi sub-network		Global Aggregated Performance
		ω_B: 1		ω_C: 0.5		
		Fitness (F)	μ	Fitness (F)	μ	
ca_4	0.61	0.99	0.00	0.62	1.48	0.61
ca_2	0.52	0.73	0.32	0.96	0.00	0.57
ca_5	0.53	0.52	0.62	0.96	0.00	0.52
ca_1	0.70	0.24	1.51	1.00	0.00	0.42
ca_3	0.81	0.00	2.57	1.00	0.00	0.23

Table 7.8: Summary of the final results of the proposed multiagent approach

In order to validate the results obtained from the proposed multi-agent system so far, the same case study (the Riyadh traffic network) has been simulated as one large traffic network using the METANET simulation model and the same traffic state has been considered. METANET has then been run to predict the performance of the five control actions ca_1 , ca_2 , ca_3 , ca_4 and ca_5 . The output shows that the METANET simulation model indicates the same trend as the proposed multi-agent system, when it recommends ca_4 as the best global control action. However, it is noted that the proposed multi-agent system is more manageable for a large network. The weights (ω_j), which represent the relative importance of sub-networks, can be changed on-line, depending on the current traffic management policy and on other considerations. Furthermore, the proposed multi-agent system is much faster than the simulation model when it is used to rank several control actions according to their approximate performance, as all training processes of the ITC-DSS and the creation of CA_{table} are done off-line. This example demonstrates that the proposed system can quickly and roughly rank several control actions based on their global performance.

7.4.3 Different Traffic Scenarios

After demonstrating the technical feasibility of the proposed multi-agent approach for identifying the best global control action, this section demonstrates the capability of the proposed multi-agent approach with different traffic scenarios. Also this part of the experiment shows how the weights (ω_j) play an important role for the operator in identifying the best global control action. In this experiment we have increased the number of control actions to ten controls and used the proposed multi-agent system to predict their global performance for five different traffic scenarios. Table 7.9 shows a brief description of the ten control actions considered in this part of the experiment, while Table 7.10 displays CA_{table} for the King Fahad sub-network with these control actions. Since the previous section showed the application of all steps of the proposed multi-agent system, the rest of this section only discusses the final results of the proposed multi-agent system for each traffic scenario.

Control Action	Description
ca_0	Doing nothing
ca_1	Using VMS at point *A* to direct traffic that goes to *D2* to use the Olaya highway.
ca_2	Using VMS at point *A* to direct traffic that goes to *D1* to use the Olaya highway.
ca_3	Using VMS at point *A* to direct traffic that goes to *D1& D2* to use the Olaya highway.
ca_4	Using VMS at point *A* to direct traffic that goes to *D2* to use the Takhassusi highway
ca_5	Using VMS at point *A* to direct traffic that goes to *D1* to use the Olaya highway and traffic that goes to *D2* to use the Takhassusi highway.
ca_6	Using VMS at point *A* to direct traffic that goes to *D2* to use the Olaya highway & on ramp metering at point *B.*
ca_7	Using VMS at point *A* to direct traffic that goes to *D1* to use the Olaya highway and applying Lane Closure on points *C* & *D*.
ca_8	Shoulder Lane Opening on section (*S1 – S2*) on the King Fahad highway and on ramp metering at points *B, C & D.*
ca_9	Using VSL at point *A* to reduce the speed limit from 90 k/h to 60 k/h, and Shoulder Lane Opening on section *(S1 – S2)* on the King Fahad highway.

Table 7.9: Brief description of the ten control actions considered in this part of the experiment.

Control Actions	Affected sub-networks				
	Olaya sub-network (Agent *B*)			Takhassusi sub-network (Agent *C*)	
	Y_{B}^{i}	R_{B}^{i}		Y_{C}^{i}	R_{C}^{i}
		L1	L2		
ca_0	0%	0%	0%	0%	0%
ca_1	+207.6%	0%	0%	-24.2%	0%
ca_2	+76.3%	0%	0%	0%	0%
ca_3	+283.9%	0%	0%	-24.2%	0%
ca_4	-8.7%	0%	0%	+225.1%	0%
ca_5	+68%	0%	0%	+255.1%	0%
ca_6	+76.3%	30%	0%	0%	0%
ca_7	+207.6%	0%	25%	-24.2%	25%
ca_8	0%	30%	30%	0%	30%
ca_9	0%	0%	0%	0%	0%

Table 7.10: The $CA_{\boldsymbol{table}}$ for the King Fahad sub-network with the ten control actions.

From Tables 7.9 and 7.10, it can be seen that control actions ca_0 and ca_9 are local control actions because they do not have any impact on the traffic state in the Olaya and Takhassusi sub-networks. Also we can observe that the average traffic demand (Y_{j}^{i}) of the Olaya sub-network is affected negatively by the control actions $ca_{1,}^{A}ca_2$, ca_3 , ca_5 , ca_6 and ca_7 , while only the control actions ca_6 , ca_7 and ca_8 reduce the outflow capacity of the Olaya sub-network out links. On the other hand, only ca_4 and ca_5 have negative impacts on the average traffic demand (Y_j^i) of the Takhassusi sub-network, while the outflow capacity of the sub-network out links are only reduced by ca_7 and ca_8 .

The five traffic scenarios considered in this experiment represent five different traffic states in the King Fahad, Olaya and Takhassusi sub-networks. Figures 7.5-7.9 show the five scenarios:

1) The first scenario represents a smooth traffic state in all sub-networks (i.e. there are no traffic incidents in any of the sub-networks).

2) The second scenario represents a congestion traffic state in all sub-networks (i.e. there are traffic incidents in all sub-networks simultaneously).

3) In the third scenario, we suppose a traffic incident in King Fahad Sub-network only.

4) The fourth scenario shows a traffic state with two simultaneous traffic incidents in the King Fahad and Olaya sub-networks.

5) The fifth scenario shows two simultaneous traffic incidents in the King Fahad and Takhassusi sub-networks.

Note that in scenarios 1 and 2, the weights (ω_j) will mainly affect the process of ranking the control actions, because the overall traffic state in the Olaya and the Takhassusi sub-networks are similar.

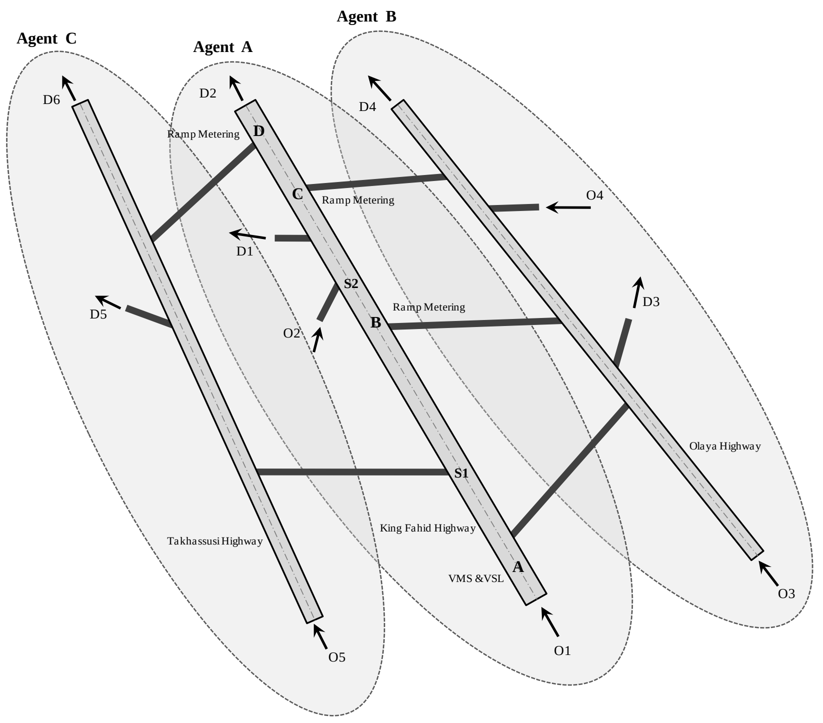


Figure 7.5: The first traffic scenario for the three traffic sub-networks.

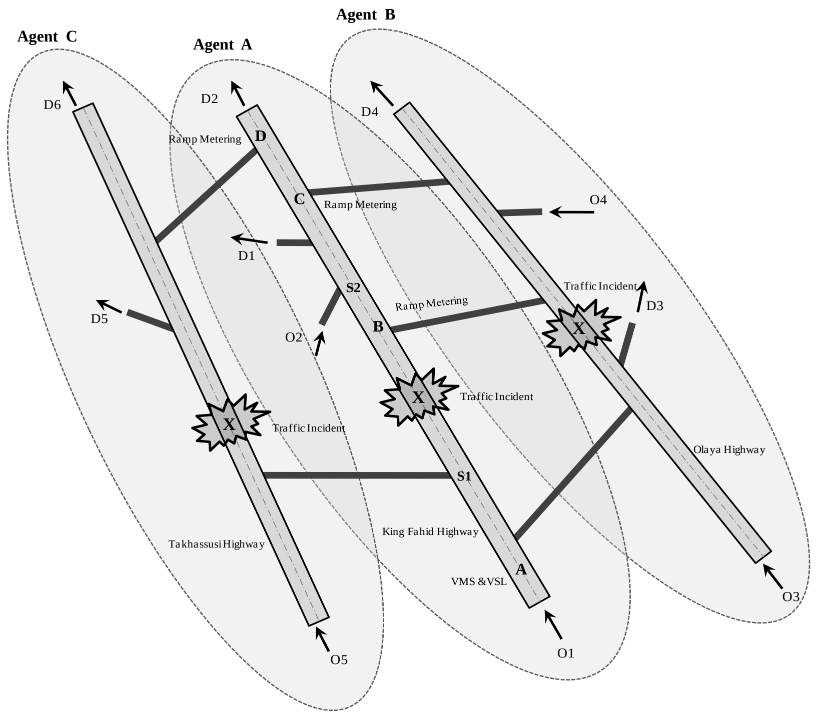


Figure 7.6: The second traffic scenario for the three traffic sub-networks.

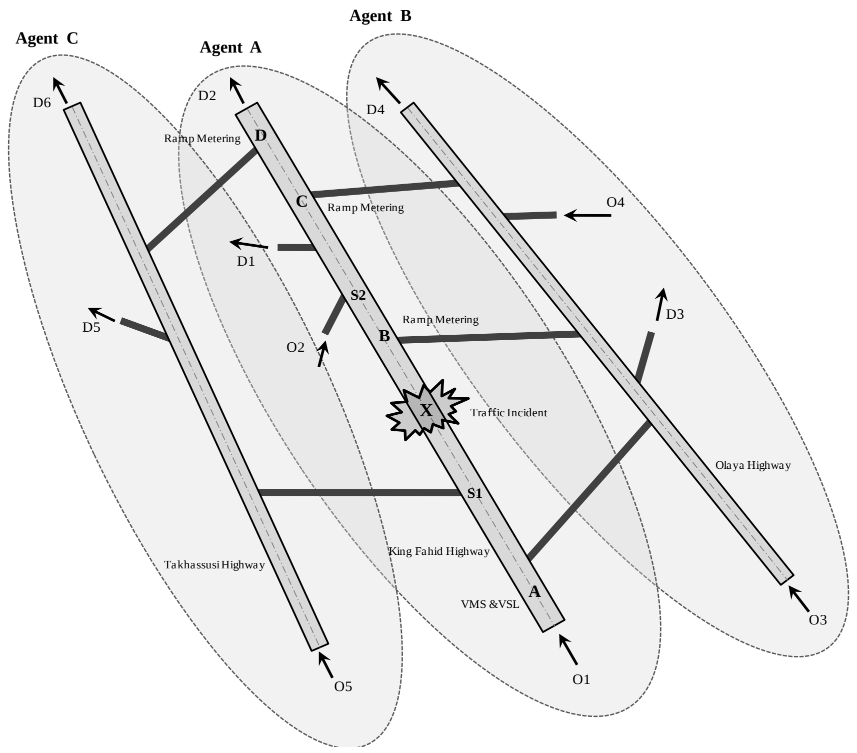


Figure 7.7 The third traffic scenario for the three traffic sub-networks.

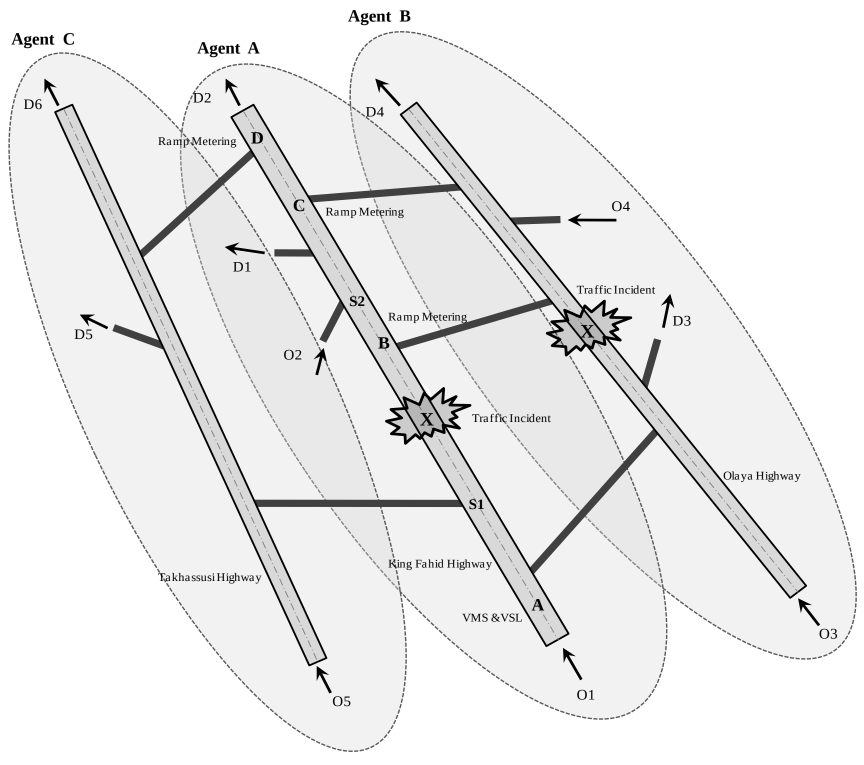


Figure 7.8: The fourth traffic scenario for the three traffic sub-networks.

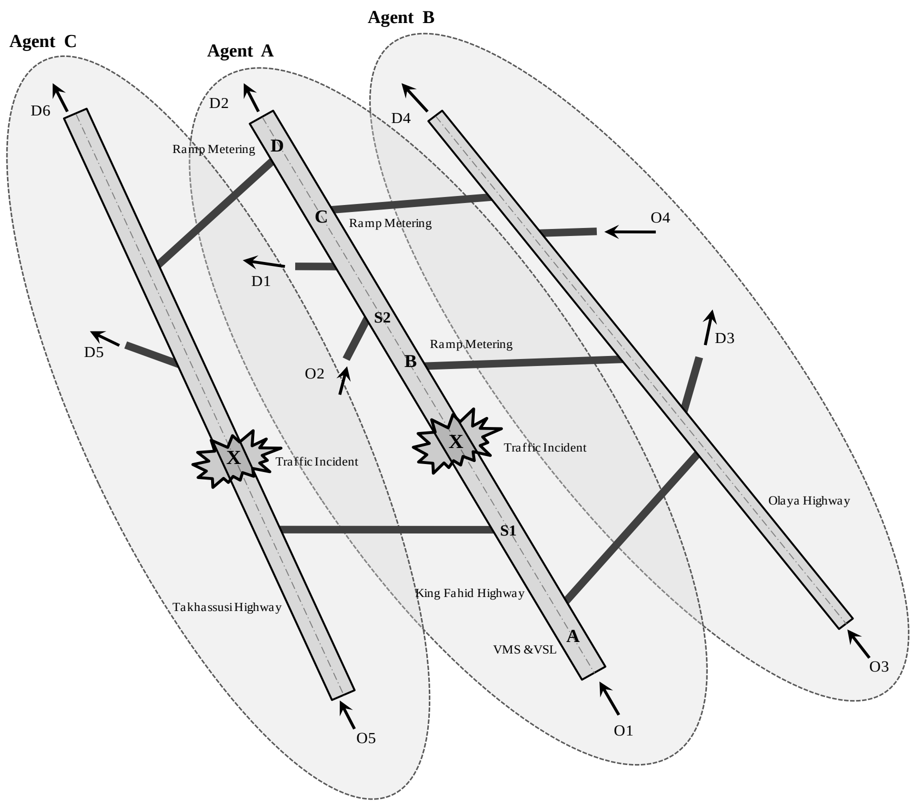


Figure 7.9: The fifth traffic scenario for the three traffic sub-networks

First Scenario:

Table 7.11 summarizes the final results obtained by the proposed multi-agent system for the first scenario. Since the first scenario represents a smooth traffic state in all sub-networks, including the King Fahad sub-network, it is observed from Table 7.11 that the global aggregated performance of the control actions are almost similar to the local aggregated performance. Moreover, since the King Fahad sub-network does not have any traffic congestion, ca_0 has high local and global performance, indicating that there is no need for applying any control action. However some control actions can cause adverse effects such as ca_4 and ca_5 . Table 7.11 also shows that all sub-networks in this scenario have the same importance ($\omega_j = 1$). The best global control action in the first scenario is ca_8 , because it further improves the traffic state of the King Fahad sub-network by opening shoulder lanes, and its side effect is not felt by the affected agents (*B* and *C*).

Control Action	Local Aggregated Performance (King Fahad sub-network)	Olaya sub-network		Takhassusi sub-network		Global Aggregated Performance
		ω_B: 1		ω_C: 1		
		Fitness (*F*)	μ	Fitness (*F*)	μ	
ca_0	0.94	0.99	0.00	0.98	0.00	0.94
ca_1	0.90	0.82	1.51	1.00	0.00	0.85
ca_2	0.91	0.96	0.32	0.98	0.00	0.93
ca_3	0.87	0.76	2.57	1.00	0.00	0.79
ca_4	0.78	1.00	0.00	0.79	1.48	0.79
ca_5	0.76	0.97	0.27	0.79	1.48	0.80
ca_6	0.92	0.94	0.62	0.98	0.00	0.93
ca_7	0.91	0.73	1.63	1.00	0.25	0.82
ca_8	1.00	0.99	0.30	0.98	0.30	0.99
ca_9	0.92	0.99	0.00	0.98	0.00	0.92

Table 7.11: Summary of the final results of the proposed multi-agent system for the first scenario.

Figure 7.10 shows a comparison between the local and global aggregated performance of the ten control actions for the first scenario. The first three bars represent the local performance (fitness) of a control action for the three sub-networks (the King Fahad sub-network, the Olaya sub-network, and the Takhassusi sub-network), while the last bar (the black one) represents the global performance of the control action.

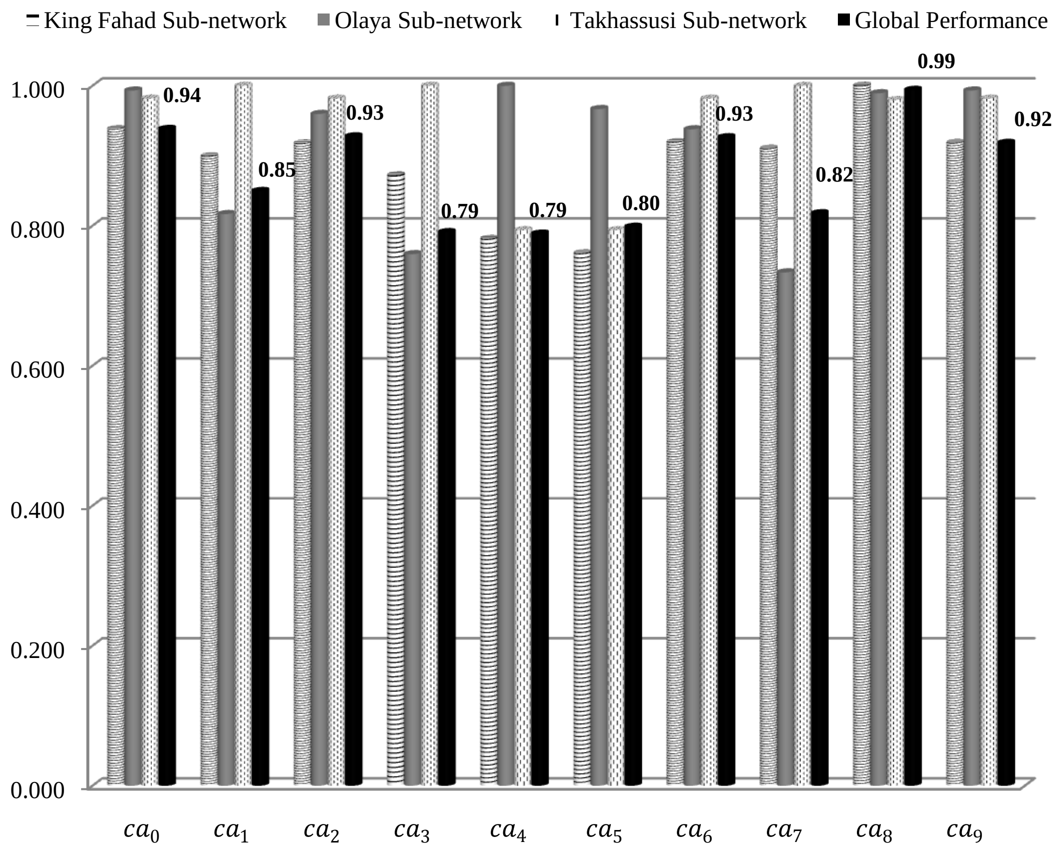


Figure 7.10: The local and global performance of the ten control actions for the first scenario.

Second Scenario:

Table 7.12 demonstrates the output of the proposed multi-agent system for the second scenario, where all sub-networks have incidents. It is observed from the table that all control actions that negatively affect a sub-network are given a low fitness by that sub-network. This is simply because all sub-networks have traffic congestion and they try to avoid any more aggravation. For example, the Olaya sub-network gives the control action ca_3 the lowest fitness (0.07) because ca_3 directs all traffic that goes to *D1* and *D2* to use the Olaya sub-network, while ca_4 has the highest fitness (0.69), because it does not have any negative impact. In addition, it reduces the traffic inflow to the Olaya sub-network by 9% (see Table 7.9). In this scenario the Olaya sub-network is assigned a larger weight ($\omega_j = 1$) than the Takhassusi sub-network ($\omega_j = 0.5$), indicating the high importance of that part of the network at that time. Thus, the output ranked list is mainly affected by the fitness obtained from the Olaya sub-network. For example, although control action ca_3 has a high local fitness (0.87) and it is recommended by the Takhassusi sub-network (with fitness 0.82), it has very low global performance (0.29) because it is given a very low fitness (0.07) by the Olaya sub-network.

Control Action	Local Aggregated Performance (King Fahad sub-network)	Olaya sub-network		Takhassusi sub-network		Global Aggregated Performance
		$\boldsymbol{\omega_B}$: 1		$\boldsymbol{\omega_C}$: 0.5		
		Fitness (*F*)	**μ**	Fitness (*F*)	**μ**	
ca_0	0.08	0.68	0.00	0.78	0.00	0.08
ca_1	0.78	0.24	1.51	0.82	0.00	0.46
ca_2	0.31	0.54	0.32	0.78	0.00	0.37
ca_3	0.87	0.07	2.57	0.82	0.00	0.29
ca_4	0.56	0.69	0.00	0.08	1.48	0.36
ca_5	0.82	0.56	0.27	0.08	1.48	0.51
ca_6	0.31	0.49	0.62	0.78	0.00	0.38
ca_7	0.78	0.24	1.63	0.82	0.25	0.46
ca_8	0.25	0.68	0.30	0.72	0.30	0.38
ca_9	0.20	0.68	0.00	0.78	0.00	0.20

Table 7.12: Summary of the final results of the proposed multi-agent system for the second scenario.

The best global control action in the second scenario is ca_5 , because it has high local performance and its negative side effects on the Olay sub-network is relatively not very high. Figure 7.11 shows a comparison between the local and global aggregated performance of the ten control actions for the second scenario.

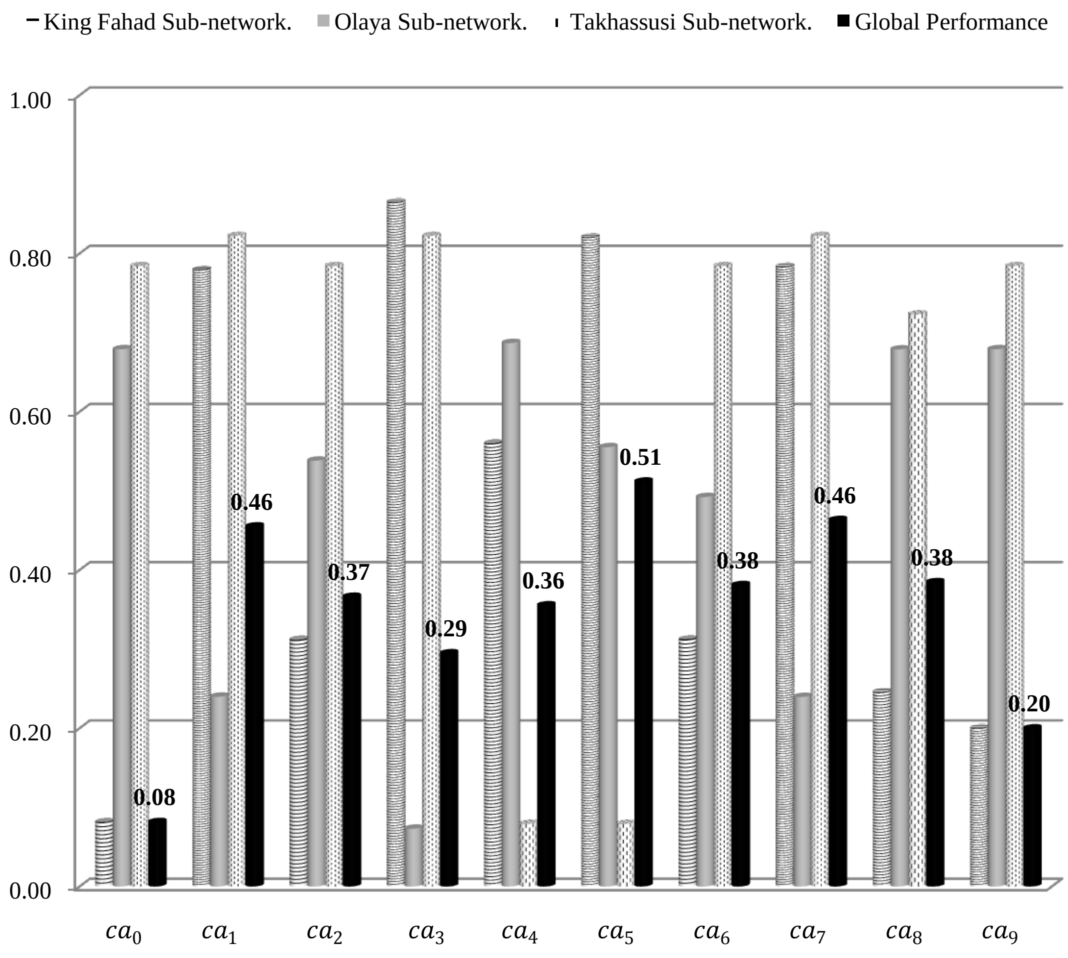


Figure 7.11: The local and global performance of the ten control actions for the second scenario.

Third Scenario:

The third scenario indicates a traffic incident in the King Fahad Sub-network, and a relatively low but equal weight ($\omega_j = 0.5$) for both sub-networks (Olaya and

Takhassusi), to indicate that solving the traffic incident in the King Fahad sub-network has a high importance at that time. Thus, Table 7.13 shows that all control actions that have a high local performance (i.e. given a high fitness by the King Fahad sub-network) have also a high global performance. In this case, the control action ca_3 is the best local and global one. Figure 7.12 shows a comparison between the local and global aggregated performance of the ten control actions for the third scenario.

Control Action	Local Aggregated Performance (King Fahad sub-network)	Olaya sub-network		Takhassusi sub-network		Global Aggregated Performance
		ω_B: 0.5		ω_C: 0.5		
		Fitness (F)	μ	Fitness (F)	μ	
ca_0	0.08	0.99	0.00	0.98	0.00	0.08
ca_1	0.78	0.89	1.51	1.00	0.00	0.83
ca_2	0.31	0.97	0.32	0.98	0.00	0.40
ca_3	0.87	0.86	2.57	1.00	0.00	0.86
ca_4	0.56	1.00	0.00	0.79	1.48	0.66
ca_5	0.82	0.98	0.27	0.79	1.48	0.82
ca_6	0.31	0.96	0.62	0.98	0.00	0.47
ca_7	0.78	0.86	1.63	1.00	0.25	0.83
ca_8	0.25	0.99	0.30	0.98	0.30	0.42
ca_9	0.20	0.99	0.00	0.98	0.00	0.20

Table 7.13: Summary of the final results of the proposed multi-agent system for the third scenario.

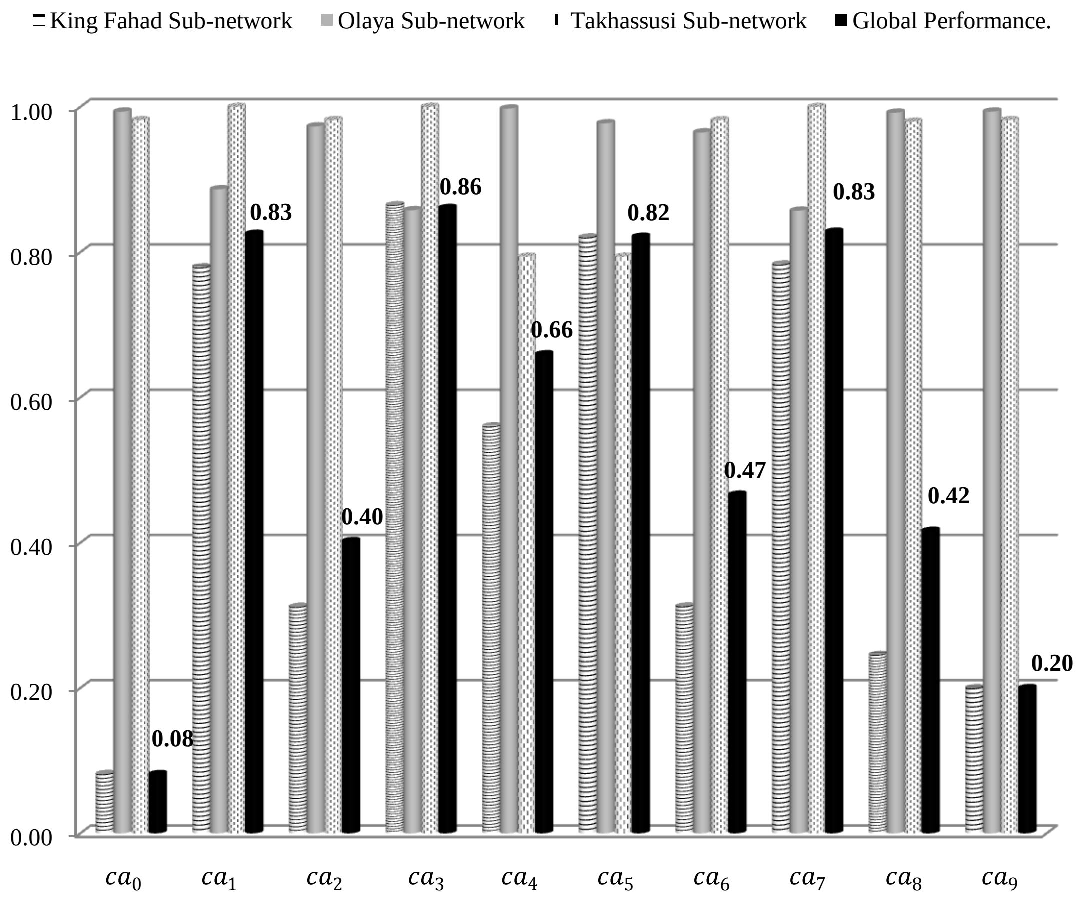


Figure 7.12: The local and global performance of the ten control actions for the third scenario.

Fourth Scenario:

In the fourth scenario, all sub-networks are assigned the same weight ($\omega_j = 1$) to show how the proposed system can recommend the best control action that improves the overall traffic state in the network. Table 7.14 summarizes the results of the fourth scenario, where both the King Fahad and the Olay sub-networks have traffic incidents. As can be seen from Table 7.14, although ca_1 and ca_3 have a high local performance (0.78 and 0.87 respectively) to solve the traffic incident in the King Fahad sub-network, they have a low global performance (0.42 and 0.24 respectively) because these control actions increase the severity of the traffic incident in the Olaya sub-network, which does not improve the overall performance of the network. Control action ca_5 is the best global control action in the fourth scenario, because it has high local performance and its negative side effect is relatively not very high on the Olay sub-network. Figure 7.13 shows a comparison between the local and global aggregated performance of the ten control actions for the fourth scenario.

Control Action	Local Aggregated Performance (King Fahad sub-network)	Olaya sub-network		Takhassusi sub-network		Global Aggregated Performance
		ω_B: 1.0		ω_C: 1.0		
		Fitness (F)	μ	Fitness (F)	μ	
ca_0	0.08	0.66	0.00	0.98	0.00	0.08
ca_1	0.78	0.18	1.51	1.00	0.00	0.42
ca_2	0.31	0.49	0.32	0.98	0.00	0.35
ca_3	0.87	0.00	2.57	1.00	0.00	0.24
ca_4	0.56	0.65	0.00	0.79	1.48	0.70
ca_5	0.82	0.51	0.27	0.79	1.48	0.78
ca_6	0.31	0.43	0.62	0.98	0.00	0.36
ca_7	0.78	0.18	1.63	1.00	0.25	0.46
ca_8	0.25	0.64	0.30	0.98	0.30	0.46
ca_9	0.20	0.64	0.00	0.98	0.00	0.20

Table 7.14: Summary of the final results of the proposed multi-agent system for the fourth scenario.

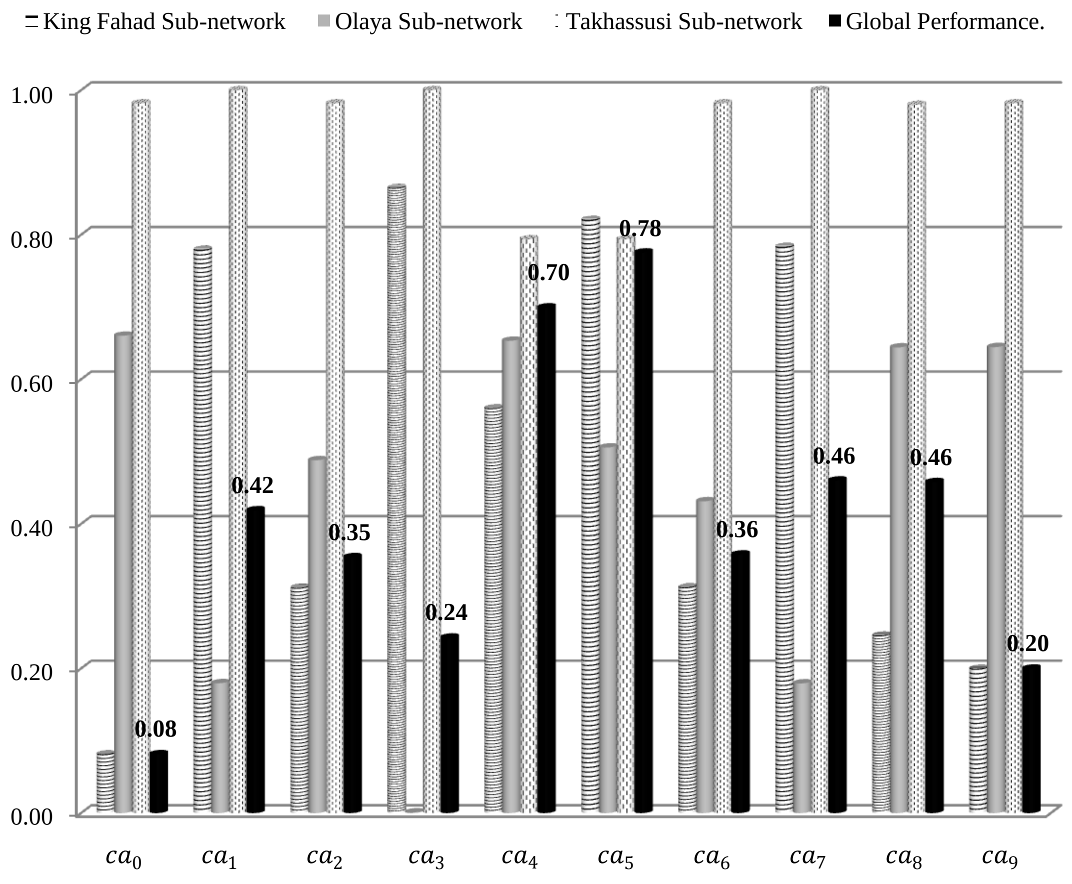


Figure 7.13: The local and global performance of the ten control actions for the fourth scenario.

Fifth Scenario:

The fifth scenario also uses a same weight$(\omega_j = 1)$ for all subnetworks, as in the fourth scenario. The fifth scenario represents the traffic state where both the King Fahad and the Takhassusi sub-networks have traffic incidents simultaneously. The final results obtained by the proposed multi-agent system for the fifth scenario are summarized in Table 7.15. The results show that ca_4 and ca_5 are not recommended as global control actions (with 0.27 and 0.44 respectively), because they have a negative effect on the Takhassusi sub-network which does not improve the overall performance of the network. Since, the Olay sub-network has not any traffic problems at this time, the best local control action ca_3 (with 0.87), which negatively affects the Olay sub-network and positively affects the Takhassusi sub-network (see Table 7.10), is selected by the proposed multi-agent system as the most appropriate global control action (with 0.86). Figure 7.14 shows a comparison between the local and global aggregated performance of the ten control actions for the fourth scenario.

Control Action	Local Aggregated Performance (King Fahad sub-network)	Olaya sub-network		Takhassusi sub-network		Global Aggregated Performance
		$\boldsymbol{\omega_B}$: 1.0		$\boldsymbol{\omega_C}$: 1.0		
		Fitness (F)	$\boldsymbol{\mu}$	Fitness (F)	$\boldsymbol{\mu}$	
ca_0	0.08	0.99	0.00	0.78	0.00	0.08
ca_1	0.78	0.89	1.51	0.82	0.00	0.84
ca_2	0.31	0.97	0.32	0.78	0.00	0.47
ca_3	0.87	0.86	2.57	0.82	0.00	0.86
ca_4	0.56	1.00	0.00	0.08	1.48	0.27
ca_5	0.82	0.98	0.27	0.08	1.48	0.44
ca_6	0.31	0.96	0.62	0.78	0.00	0.56
ca_7	0.78	0.86	1.63	0.82	0.25	0.83
ca_8	0.25	0.99	0.30	0.78	0.30	0.49
ca_9	0.20	0.99	0.00	0.78	0.00	0.20

Table 7.15: Summary of the final results of the proposed multi-agent system for the fifth scenario.

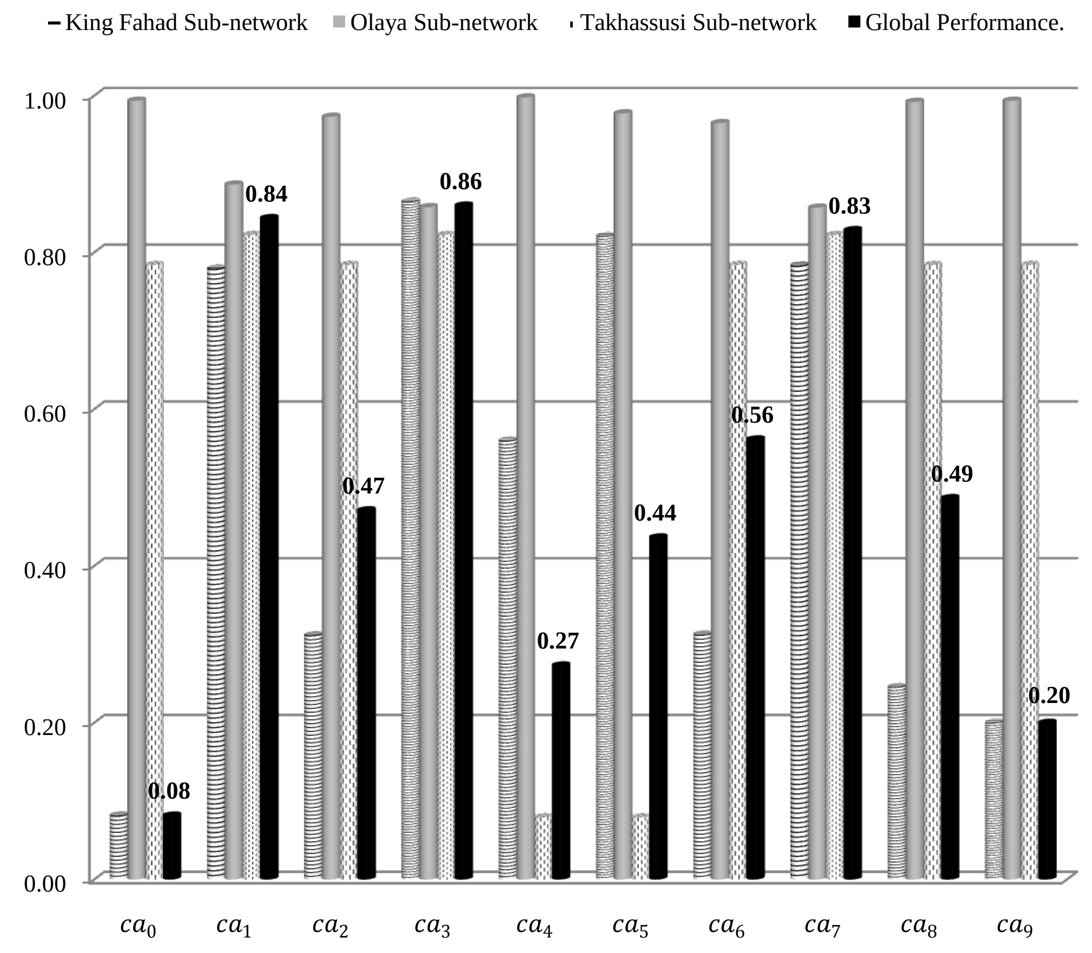


Figure 7.14: The local and global performance of the ten control actions for the fifth scenario.

7.4.4 Cross-Scenario Analysis

In the previous sub section five different traffic scenarios were considered to represent five different traffic states in the King Fahad, Olaya and Takhassusi sub-networks and ten traffic control actions with different impacts on the traffic state of those sub-networks were proposed. According to Tables 7.9 and 7.10 the traffic control actions considered in this experiment can be classified based on their impact into five groups: 1) internal control actions that do not have any impact on the traffic state in the Olay and Takhassusi sub-networks (ca_0 and ca_9); 2) control actions that affect the traffic states in the Olay and Takhassusi sub-networks (ca_5, ca_7 and ca_8); 3) control actions that affect only the traffic state in the Olay sub-network (ca_1, ca_2, ca_3 and ca_6) 4) control actions that affect only the traffic state in the Takhassusi sub-network (ca_4). In this section we discuss and analyze the results obtained from the five traffic scenarios.

In general, the results of the traffic scenarios demonstrates that the global performances of the traffic control actions are not fixed in all five traffic scenarios but fluctuate according to the traffic states in all sub-networks. For example, the best global traffic control actions in the first scenario, when the traffic state in all sub-networks was a smooth (i.e. there are no traffic incidents in any of the sub-networks), is different from the best ones in the second scenario, when there is a congestion traffic state in all sub-networks. The global performance of each control action is influenced by its local performance (King Fahd traffic state) and the traffic state in the affected sub-network(s).

For example, the global performances of the control actions ca_1, ca_2, ca_3 and ca_6 are influenced by their local performances and the traffic states in the Olay and Takhassusi sub-networks, while the global performances of the control actions ca_4 is influenced by its local performances and the traffic states in the Takhassusi sub-network only. The global performances of the internal control actions (i.e. ca_0 and ca_9) are only influenced by King Fahd traffic state, so in the second, third, fourth and fifth scenarios, where the same traffic state of King Fahad subnetwork was considered, the global performances of those control actions were still the same.

Local and Global aggregated performances:

It is observed from the analysis of the results of the five traffic scenarios that there are two different aggregated performances for each traffic control action: the local aggregated performance and the global aggregated performance. The local aggregated performance of a control action represents its local impact on the King Fahad sub-network only. I.e. it is the fitness of a control action, which is provided by agent *A*, to solve the current traffic congestion in the King Fahad sub-network. For calculating the local aggregated performance of a traffic control action (as explained in Chapter 4), only the traffic state of the King Fahad subnetwork is considered without taking into account any change that may happen in the traffic flow of the affected subnetworks due to the application of that traffic control action.

Therefore, it is observed from Tables (7.12-7.15) that the control actions with the higher local aggregated performance are the control actions that mainly reduce the traffic congestion severity in the King Fahad subnetwork. For example, in the second scenario, the control actions ca_3 and ca_5 have the higher local aggregated performance (0.87 and 0.82) because they pass the most of the traffic inflow of the King Fahad subnetwork to the other subnetworks (Olaya and Takhassusi) which subsequently reduce the traffic congestion in the King Fahad subnetwork, while the control action ca_0 has the lower local aggregated performance because ca_0 does improve the traffic state in the King Fahad subnetwork.

The global aggregated performance of a control action represents how good (or bad) the control action to solve the current traffic congestion in the King Fahad subnetwork with taking into consideration the traffic state on the network level (i.e. its performance on all subnetworks). As explained in the previous sections in this chapter, the traffic state of all affected sub-networks and the change that may happen in the traffic flow of those sub-networks due to the application of a traffic control are considered in the calculation of the global aggregated performance of that control action. So, it is observed from the analysis of the results of the five traffic scenarios that the fitness of a control action provided by the affected sub-networks influences the global aggregated performance of that control action. For example, in the fourth scenario, when the Olay sub-network has traffic congestion, the Olay sub-network's fitness of the control actions ca_1 , ca_3 and ca_7 is very low (0.18, 0.00 and 0.18 respectively), because those control actions have negative impacts on the Olay sub-network. Subsequently, the global aggregated performances of those control actions are low (0.42, 0.24 and 0.46 respectively), although their local aggregated performance are high (0.78, 0.87 and 0.78 respectively).

ω_j and μ_j^i parameters

There are two weighting parameters (ω_j and μ_j^i) are used in the calculation of the global aggregated performances of the traffic control actions (see Equations 7.2 and 7.3). The results analysis of the five traffic scenarios shows how these weighting parameters play an important role in ranking the control actions in each scenario. The ω_j parameter represents the relative contribution of sub-network j to the global aggregated performances of the traffic control actions. For example, in the second scenario, when the Olaya sub-network was assigned a larger weight ($\omega_B = 1$) than the Takhassusi sub-network ($\omega_C = 0.5$), indicating the high importance of that part of the network at that time, the contribution of the Takhassusi sub-network (i.e. the fitness of the traffic control actions provided by the Takhassusi sub-network) had less impact on the global aggregated performances than the contribution of the Olaya sub-network (i.e. the fitness of the control actions provided by the Olaya sub-network). The ω_j parameter has been employed in the calculation of the global aggregated performances of the control actions in the proposed multi-agent approach to increase the flexibility by allowing the operator, depending on the current traffic management policies and other consideration, to increase or decrease the relative importance (contribution) of a part of the network.

The μ_j^i parameter expresses how much impact the traffic control action ca_i has on the affected sub-network *j*. When μ_j^i is low, the global aggregated performances of control action *i* is meanly affected by the contribution (fitness) of sub-network *j*. In contrast, when μ_j^i is high, the global aggregated performances of control action *i* is mainly affected by the contribution (fitness) of sub-network *j*. Also, μ_j^i is used, when equals zero, to indicate irrelevant sub-networks. By using μ_j^i, we ensure that only the mainly affected sub-networks are more considered in the identification of the global aggregated performances of the control actions.

For example, since the internal control actions (ca_0 and ca_9) do not have any impact on the Olay and Takhassusi sub-networks, the associated μ_B^i and μ_C^i are assigned zeros to disregard the contributions (fitness) of these sub-networks in the global aggregated performances of ca0 and ca9. While in the control actions (ca_1 , ca_2 , ca_3 and ca_6), which affect only the traffic state in the Olay sub-network, the associated μ_C^i are assigned zeros to only consider the contribution of the Olay sub-network, and the opposite in ca_4 .

Based on this analysis of the results obtained from the five traffic scenarios, it can be concluded that the proposed multi-agent approach can be effectively used in ranking a number of control actions based on their global performance in different traffic states with different traffic management policies. It is observed how the proposed approach efficiently consider the current traffic states of all parts of the network (i.e. King Fahad, Olaya and Takhassusi sub-networks) in the process of identifying the global performances of the control actions, and also how the proposed weighing parameters (ω_j and μ_j^i) play an effective role in this process.

7.5 Summary

In this chapter, we have extended the intelligent traffic control decision support system (ITC-DSS) presented in the previous chapters, which has previously been used only for controlling a small-sized network, to be used to control a large-sized network. We have opted for a multi-agent approach where the total network was divided into a number of sub-networks, each of which has its own agent. The coordination between those agents was achieved through a high level agent called a coordinator, which receives proposed local control actions from the agent of the sub-network with regard to an incident, resolves conflicts between other affected agents, and sends the globally acceptable solution back to that agent. The chapter presented the proposed multi-agent approach, including its structure and all operational steps.

In order to test the technical feasibility of the proposed multi-agent decision support system, a case study of a large section of the ring-roads around Riyadh, with five different traffic states has been presented and discussed. The obtained results demonstrated the capabilities of the proposed multi-agent system in order to help the traffic centre operator to identify the best global control action for a large traffic network. The research results of this chapter have been partially reported in [2].

Chapter 8

Conclusions and Future Work

8.1 Conclusions

In this thesis we have proposed and analysed the application of an Intelligent Traffic Control Decision Support System (ITC-DSS) for road traffic management to assist the human operator to manage the current traffic state in real-time.

An investigation into the real-time intelligent decision support system using a combination of three soft-computing approaches, namely fuzzy logic, neural networks, and a genetic algorithm have been carried out to demonstrate the merits of the proposed system through a set of experiments and case studies. In Chapter 4, we introduced the structure of ITC-DSS which employs FNN-Tool that combines the capabilities of fuzzy reasoning in measuring imprecise and dynamic factors, and the capabilities of neural networks in learning from processes. The system has been successfully trained and subsequently tested for a traffic case-study of a small section of the road traffic network in Riyadh city in Saudi Arabia. The results obtained are promising and demonstrate that the proposed intelligent decision support system can provide an effective support for real-time traffic control. A comparison between the results of the proposed system and the results obtained by the traffic simulation model (METANET) confirmed the validity of the proposed intelligent real-time traffic control decision support in predicting the control action performance and clearly showed the ability of the proposed system in terms of processing speed and flexibility.

The experimental results described in Chapter 4 demonstrated that the parameters w_{C_d} , C_d^{min} and C_d^{max} , (where w_{C_d} is the weight of the performance criterion C_d, and C_d^{min} and C_d^{max} are the minimum and the maximum values of C_d) play an important role in the calculation of the aggregated performance of a control action. Tuning them by the operator is a matter of describing the desired behaviour of the traffic state.

In this thesis, we recognise the well-known facts that employing an effective learning process can lead to a good performance on the part of a fuzzy neural network. Therefore we have investigated the development of an effective learning process for our fuzzy neuronal network (FNN-Tool). In Chapters 5 and 6, we have introduced an effective learning approach for the FNN-Tool, consisting of three stages. The first stage initializes the membership functions using a self-organization algorithm. The second stage identifies fuzzy rules using a genetic algorithm (GA) based learning method. The third stage employs the back-propagation neural network algorithm for fine tuning the system parameters.

In Chapter 5, a simple and effective GA-based fuzzy rule identification method was developed and tested in order to be used in the second stage of the learning process of the FNN-Tool. In our developed fuzzy rule identification method, the fine tuning of the fuzzy rules weight is done in a separate learning stage (stage three). We used an integer representation (encoding) of the problem which reduced the length of the chromosome (compared with binary representation) as well as the size of the GA search space. We showed that the developed GA-based fuzzy rule identification method was able to identify all the relevant fuzzy rules correctly.

In Chapter 6, the performance of the proposed three-stage learning approach was evaluated using a well-known benchmark example (i.e. the Box–Jenkins time series). Moreover, the prediction capability of the FNN-Tool trained by the proposed learning approach, was assessed for predicting the performance of traffic control actions on a given traffic state. The results obtained have demonstrated the ability of the proposed learning approach to identify all relevant fuzzy rules from the training data. Furthermore, a comparative analysis has demonstrated that our approach leads to better performance than that of other well-known approaches. The main features and advantages of the FNN-Tool and the three stage-based learning approach proposed in this thesis are the use of GA for extracting fuzzy IF-THEN rules from numerical information, and the effective tuning process where it is easy to modify the fuzzy rule base and the structure of the FNN-Tool when new data (or linguistic information) becomes available.

In order to evaluate the impact of different parameters and operations on the GA performance, sensitivity analysis on crossover probability and type, mutation probability, and population size has been carried out in Section 6.4.3. The obtained results demonstrate that too low crossover probability might lead to a bad performance of GA, especially with one-point crossover. Also, the GA performance was not good if too low (e.g. 0.001 and 0.005) or too high (e.g. 0.2) mutation probabilities are used.

In Section 6.4.4, two experiments were run. In the first experiment, the BP algorithm was employed only to adjust the membership parameters (centres, widths), while in the second experiment the membership parameters and the fuzzy rules weight were adjusted. The experimental results show that the error rate can be improved by learning the weights of the fuzzy rules.

It is observed that the proposed ITC-DSS has the potential to be a useful supporting tool for controlling a small traffic network. However, ITC-DSS does not scale well with a large traffic network. In order to scale up the proposed ITC-DSS to be used for controlling a large traffic network, in Chapter 7 we developed our ITC-DSS using a multi-agent approach. In the multi-agent approach we divided the large traffic network into several (possibly overlapping) sub-networks, each of which has its own ITC-DSS and own agent. To make sure that all available control actions serve the same objective (i.e. the most appropriate global control action), we used a coordinative control strategy through a high level agent called a coordinator. The coordinator receives proposed local control actions from the agent of the sub-network with regard to an incident, resolves conflicts between other affected agents, and sends the globally acceptable solution back to that agent.

The technical feasibility of the proposed multi-agent approach was tested for a case study involving a large section of the traffic network in the city of Riyadh in Saudi Arabia, and the output was compared with the output of the simulation model METANET. The results showed that the proposed multi-agent approach is more manageable and much faster than the simulation model when it is used to rank several control actions according to their approximate performance. We also showed the capability of the proposed multi-agent approach for controlling large networks through different traffic scenarios with different traffic policies.

The experimental results in Section 7.5.3 demonstrated how the weights (ω_j), which represents the relative importance of the sub-networks and the parameters (μ_j), which express how much impact the traffic control action ca_i^A has on the affected sub-network, plays an important role for the operator in identifying the best global control action.

We believe that the intelligent traffic control decision support system developed in this thesis has the potential to be a useful intelligent tool for assisting the human operator in the traffic control centre to optimally manage the current traffic state in the real-time. The main advantages of our intelligent traffic control decision support system is that it is easy to use, reliable and faster. Moreover, it is simple to be developed and implemented.

8.2 Usability Issues

Human computer interaction (HCI) is an important area of computer science, which involves the interaction of human user, task, and computer. HCI combines the physical, logical, conceptual, and language-based interactions between the human user and the computer for achieving some goals [81]. The basic goal of HCI, (as defined in Interacting with Computers [33]), “is to develop or improve the safety, utility, effectiveness, efficiency and usability of systems that include computers”.

Usability is an aspect of HCI. It is defined as the effectiveness, efficiency and satisfaction with which specified users can achieve specified goals in particular environments [38]. Nielsen [121] defines usability by five quality components:

Learnability: how easy users accomplish their basic tasks the first time they encounter the design.

Efficiency: how quickly can users perform a task using the interface of the system?

Memorability: how easy is to memorize how to use the interface of the system and how easily users can reuse the system after a break?

Errors: How many errors do users make using the interface of the system and how serious are these errors

Satisfaction: How do users like using the system's interface?

In order to successfully develop the proposed traffic control system, we must consider the user interface design carefully and thoroughly. The user interface should provide all necessary functions and information to the traffic operator. For example, the interface window of the proposed traffic control system that is presented to the operators should allow the operator to enter the parameters that describe the current traffic state and then should provide him/her with a ranked list of the various possible of control actions based on their performance over that current state. In order to explain why specific control action is recommended, the interface window should display for each control action its performance which is represented by some traffic criteria e.g. the total travel time, total distance traveled, total waiting time, total time in net, vehicles in net, vehicles driven in, vehicles driven out, and total fuel consumption. In addition, the interface window should allow the operator to specify the weights w_{C_d} for the various performance criteria to meet his-specified objectives.

Another example for increasing the usability of the proposed traffic control system is that the system should allow the user (expert operator) to effectively include expert knowledge within the system when it becomes available. This can be done be providing the user with expert options which give him some facilities, for example, to change the membership functions (i.e. centres c and widths σ) using linguistic information provided by experts. In addition in order to add new fuzzy rules (or delete an exciting rules), the expert options should allow the user to adjust the fuzzy rules weights (i.e. connection weights $W_{u,nm}$ of the links connecting nodes RL_u in layer 3 to $OL_{n,m}$ in Layer 4). Since the structure of FNN-Tool created with all possible fuzzy rules where fuzzy rule with weight $W_{u,nm} \neq 0$ represents a fuzzy rule to be considered and with weight $W_{u,nm} = 0$ to be ignored, assigning zero weight ($W_{u,nm} = 0$) to any considered fuzzy rule indicates that the rule is deleted. In contrast, assigning a value ($W_{u,nm} \neq 0$) to any ignored fuzzy rule indicates that the rule is added.

The main challenges for practical implementation of the research can be summarized in the following points:

- Understanding the users (traffic operators) who will use the proposed system is the first and the most important challenge for practical implementation of the research. It is necessary to understand who the user of the system is, what level of expertise he/she has, what he/she is likely to assume about the system and the environment in which he/she is operating. Golud said in his paper [47] that “designers are often reluctant to define the users and even when they have done so seem reluctant to define the users and system”. It also has to carefully examine the user expertise and background to decide what type of help facilities and training that will be needed for the support of the system. Users must understand how to express instructions to the computer, how to organize these instructions, and how the computer executes these statements [121].

- Another challenge is training the proposed system. The proposed system is FNN based system, thus its performance mainly affected by the quality of training. When the number of training pairs is small, or perhaps not representative of the possibility space, the system results are predictably poor. Therefore, it is necessary to ensure that the training data is sufficient to cover the entire problem domain, including maximum and minimum values for each variable, as well as a good distribution of values within this range. Moreover, incorporation real road traffic data is very important and needs to be considered during the training process to increase the accuracy of the proposed system.
- The hardware and software solutions that are needed to implement the proposed traffic control system are another challenge. For example, the proposed system should be tested in an advanced traffic control centre where the current traffic state is monitored online to accurately provide the proposed system with the required input information, however, based on the author experience, the current online traffic monitoring devices and incident detection methods in most of exciting traffic control centers in Saudi Arabia are still insufficient.

8.3 Future Work

This section suggests the following directions for future research arising from this thesis:

In Chapter 5, a GA-based learning method was used for identifying fuzzy rules of our FNN-Tool. In the GA-method the chromosome was encoded with all possible rules. This has the advantage of decreasing the chance to miss any relevant rule. In the experiments undertaken so far, the number of variables was limited, thus the total number of possible rules was still not extremely large. In some cases, when the number of variables and fuzzy sets are extremely large, encoding a chromosome with all possible rules is not efficient, making the length of a chromosome very large and, consequently, the whole genetic learning process becoming overloaded. Therefore, an interesting further investigation of this issue would be to use a variable length chromosome. In this case it would be interesting to employ a clustering algorithm (e.g. as detailed in [102]) just to identify an optimal number (n) (i.e. fuzzy rules) to start with. A further option that can be explored is the effectiveness of multi-objective optimization approach, where the objectives are minimizing the error level and minimizing the number of fuzzy rules.

The three-stage learning approach for training the FNN-Tool, presented in Chapter 6, is an off-line learning approach. The results obtained have verified the effectiveness of the FNN-Tool with the three-stage learning approach for predicting the performance of traffic control actions. To increase the efficiency of the FNN-Tool and to avoid a re-run of the learning process when the training set is updated, the development of on-line learning strategy for FNN-Tool deserves further investigation. In this case the membership functions parameters (centres and widths) and the fuzzy rules weights should be fine-tuned on-line according to the error between the predicted performance of the applied traffic control action obtained from the FNN-Tool and the actual performance of the applied traffic control action obtained from the monitoring system.

Another interesting future work could be an experimental analysis of the FNN-Tool's noise tolerance ability. This can verify that the FNN-Tool is able to maintain its predictive capability in terms of control actions' performance under the influence of noisy input data. In this case, a new test set should be generated by introducing white noise of a specific percentage (e.g. 10%) into the original test data. Then the new test data is used for predicting the performance of control actions using the FNN-Tool. It would be interesting to evaluate the FNN-Tool's noise tolerance ability and to compare the results with that of other models (e.g. [90] and [137]).

In Chapter 7 we proposed a multi-agent approach for road traffic control systems. Numerous areas of research arise from the multi-agent approach.

- So far we have only used the proposed multi-agent approach as a framework to scale up ITC-DSS to be used in one traffic control centre (e.g. one computer). Further work could be done to develop the multi-agent approach to be used at different traffic control centres. In this case, extensive work needs to be carried out to investigate the communications issues between agents and the coordinator.
- In the proposed multi-agent approach, a large network is divided into a num-

ber of sub-networks. The main aim of this is to minimize the number of traffic variables that are required to characterize the current traffic (i.e. the inputs of the FNN-Tool), as well as to minimize the number of possible traffic control actions that can be applied to manage the traffic state. An investigation of an optimal way to split the network can be carried out. This should take into account some additional factors such as the topology of the network, the interrelations between the traffic control actions at different locations in the network, overlapping sections between sub-networks, etc.

- For calculating the control action fitness, the affected agents run their ITC-DSS with only the internal control actions (i.e. which mainly influence the traffic flow within the sub-network) to avoid the negative knock-on effect of creating a new problem. However, this may reduce the chance of taking advantage of the external control actions (i.e. that have an effect in the entire network) that may help to yield an optimal global traffic state. It would be interesting to investigate this more extensively.
- In the proposed multi-agent approach, the coordinator receives the proposed local control actions for an agent, then communicates with the affected agent to produce the global performance for the proposed local control actions (see Chapter 7.4.2). It would be interesting to develop an intelligent decision support tool (using, for example, FNN, knowledge based systems or case-based reasoning) to help the coordinator to predict the global performance of a control action using previous history. For example, when the coordinator receives the proposed local control actions, it needs to check the traffic state of the affected agents, then use the intelligent decision support tool to predict

the global performance of the control actions. This will speed up the coordination process.

- Since the implementation of the multi-agent centralized architecture is much less complex than the decentralized architecture, the centralized architecture has been adopted in our proposed multi-agent system. Although the centralized architecture has shown to perform well, it still has some limitations comparing with decentralized architecture. First, a centralized based system is more sensitive to disruptions than a decentralized based system, e.g., when the coordinator becomes unavailable, global traffic control system breaks down. Second, the response time is potentially increased in a centralized based system. For example, when an incident happens, this must communicated all the way up to a the coordinator, after which the coordinator makes a decision and communicates it all the way down to those carrying out the work. A faster response may be obtained by allowing the agent with the incident to take immediate action. Third, the centralized architecture also suffers difficulties in scalability. The complexity of the coordination task grows exponentially in the size of traffic control agents. In addition, the adding of additional control actions (and/or control devices) is easier in a decentralized based system than in a centralized based system. Therefore, an interesting further work is to investigate the possibility of using the decentralized architecture in our proposed multi-agent approach. In this case, similarly, the network is divided into a set of overlapping sub-networks each of which has its autonomous agent. However, the centralized and decentralized approaches differ significantly in the way that theses traffic agents are coordinated. To perform the coordination task in our proposed multi-agent approach in decentralized fashion, each agent should have its own control actions data-table (CA_{table}). This control actions table should be constructed with all possible traffic control actions that can be applied on the sub-network associated with that agent. When a traffic problem is detected by the monitoring subsystem controlled by the agent, instead of contacting the coordinator as in the centralized approach, it runs its ITC-DSS to come up with a ranked list of possible local control actions. Then, similarly to what the coordinator does in the centralized approach, the agent determines the affected agents for each local control action using its CA_{table} and contact them to get the fitness of these control actions in order to calculate the aggregated global performance of the control action.

Finally, the investigation presented in this thesis has verified the effectiveness of the proposed intelligent control decision support system for road traffic management. It would be interesting to investigate the application of the proposed intelligent decision support system for another area, where real-time decision-making is involved, for examples stock trading and wastewater control.

Bibliography

[1] Almejalli K, Dahal K, Hossain A. GA-Based Learning Algorithms to Identify Fuzzy Rules for Fuzzy Neural Networks. In: the 7th International Conference on Intelligent Systems Design and Applications, IEEE Computer Science, ISDA07; 2007; Rio de Janeiro, Brazil. p. 289-296.

[2] Almejalli K, Dahal K, Hossain A. An Intelligent Multi-agent Approach for Road Traffic Management Systems. In: the 18th IEEE International Conference on Control Applications (CCA) ; 2009; Saint Petersburg, RUSSIA.

[3] Almejalli K, Dahal K, Hossain A. Real Time Identification of Road Traffic Control Measures. In: Fink A, Rothlauf F, editors. Advances in Computational Intelligence in Transport, Logistics, and Supply Chain Management. Springer; 2008. p. 63-80.

[4] Ang K, Quek C. RSPOP: Rough Set-based Pseudo-Outer-Product Fuzzy Rule Identification Algorithm. Neural Computation. 2005 Jan.;17(1):205-243.

[5] Ang K, Quek C, Pasquier M. POPFNN-CRI(S): Pseudo Outer Product Based Fuzzy Neural Network Using the Compositional Rule of Inference and Singleton Fuzzifier. IEEE Transactions on Systems, Man and Cybernetics,

Part B. 2003;33(6):838-849.

[6] Barceló J, Ferrer J. AIMSUN2: Advanced Interactive Microscopic Simulator for Urban Networks, User’s Manual. Technical report. Catalunya: Tech. rep., Universitat Politecnica de Catalunya, 1997; 1997 Universitat Politecnica de Catalunya.

[7] Barlow H. Unsupervised Learning. Neural Computation. 1989;1(3):295-311.

[8] Beauchamp-Baez G, Rodriguez-Morales E, Muniz-Marrero E. A Fuzzy Logic Based Phase Controller for Traffic Control. In: the 6th IEEE International Conference on Fuzzy Systems; 1997; Barcelona, Spain. p. 1533-1539.

[9] Bellemans T. Traffic Control on Motorways. PhD Thesis. Leuven: Catholic University of Leuven; 2003. ISBN 9056824139.

[10] Benitez J, Castro J, Requena I. Are Artificial Neural Networks Black Boxes? IEEE Transaction in Neural Networks. 1997;8:1156-1164.

[11] Berthold M, Diamond J. Constructive Training of Probabilistic Neural Networks. Neurocomputing. 1998;19:167-183.

[12] Bielli M, Ambrosino G, Boero M, editors. Artificial Intelligence Applications to Traffic Engineering. VSP International Science Publishers; 1994.

[13] Bingham E. Reinforcement Learning in Neurofuzzy Traffic Signal Control. European Journal of Operational Research. 2001;131(2):232-241.

[14] Bogenberger K, El-Araby K, Keller H. Design of a Genetic Fuzzy Approach for Ramp Metering. In: Proceedings IEEE Intelligent Transportation Systems; 2000; Dearborn, USA. p. 470-475.

[15] Bogenberger K, Keller H. An Evolutionary Fuzzy System for Coordinated and Traffic Responsive Ramp Metering. In: the 34th Annual Hawaii International Conference on System Sciences; 2001; Maui, Hawaii. p. 10-20.

[16] Box G, Jenkins G, Reinsel G. Time Series Analysis: Forecasting and Control. San Francisco: Holden-Day; 1970.

[17] Buckley J, Hayashi Y. Numerical Relationship Between Neural Networks, Continuous Functions and Fuzzy Systems. Fuzzy Sets Systems. 1993;60(1):1-8.

[18] Buckley J, Hayashi Y, Czogala E. On the Equivalence of Neural Nets and Fuzzy Expert Systems. Fuzzy Set Systems. 1993;53(2):129-134.

[19] Castro P, Camargo H. Focusing on Interpretability and Accuracy of a Genetic Fuzzy System. In: the 14th IEEE International Conference on Fuzzy Systems; 2005; Reno, Nevada (USA). p. 696-701.

[20] Chambers L. The Practical Handbook of Genetic Algorithms: Applications. 2nd ed. Boca Raton: CRC Press; 2000.

[21] Cheng X, Yang Z. Intelligent Traffic Signal Control Approach Based on Fuzzy-Genetic Algorithm. In: the Fifth International Conference on Fuzzy Systems and Knowledge Discovery, FSKD '08.; 2008; Shandong. p. 221-

225.

[22] Chen M, Wang Q, Yang Y. A Hybrid Knowledge-Based Neural-Fuzzy Network Model with Application to Alloy Property Prediction. Lecture Notes in Computer Science. 2007 July;4491:528-535.

[23] Cheu R, Ritchie S, Recker W, Bavarian B. Investigation of a Neural Network Model for Freeway Incident Detection. In: the Second International conference on the Artificial Intelligence and Civil Engineering; 1991; Edinburgh. p. 267-274.

[24] Chiu S, Chand S, Moore D, Chaudhary A. Fuzzy Logic for Control of Roll and Moment for a Flexible Wing Air Craft. IEEE Control Systems Magazine. 1991 June;11(4):42-48.

[25] Coley D. An Introduction to Genetic Algorithms for Scientists and Engineers. New Jersey: World Scientific; 1999.

[26] Cordón O, Gomide F, Herrera F, Hoffmann F, Magdalena L. Special Issue on Genetic Fuzzy Systems. Fuzzy Sets and Systems. 2004;141(1).

[27] Cuena J, Hernandez J, Molina M. An Intelligent Model of Road Traffic Management in the Motorway Network Around Barcelona. In: the 14th International Federation on Information Processing (IFIP) World Computer Congress; 1996; London. p. 173-180.

[28] Cuena J, Hernández J, Molina M. Knowledge Oriented Design of an Application for Real Time Traffic Management: The TRYS System. In:

European Conference on Artificial Intelligence (ECAI'96); 1996; Budapest.

[29] Cuena J, Hernandez J, Molina M. Knowledge-based Models for Adaptive Traffic Management Systems. Transportation Research Part C: Emerging Technologies. 1995;3(5):311-337.

[30] De Schutter B, Hoogendoorn S, Schuurman H, Stramigioli S. A Multi-Agent Case-Based Traffic Control Scenario Evaluation System. In: Proceedings IEEE Intelligent Transportation Systems.; 2003; Schangai. p. 678-683.

[31] Deb K. Multi-Objective Optimization Using Evolutionary Algorithms. 1st ed. John Wiley & Sons; 2001.

[32] Dia H. An Object-Oriented Neural Network Approach to Short-Term Traffic Forecasting. European Journal of Operational Research. 2001;131(2):253-261.

[33] Diaper D. The Discipline of Human-Computer Interaction. Interacting with Computers. 1989;1(1):3-5.

[34] Donieca A, Mandiaub R, Piechowiakb S, Espiéc S. A Behavioral Multi-agent Model for Road Traffic Simulation. Engineering Applications of Artificial Intelligence. 2008;21(8):1443-1454.

[35] Dreyfus G. Neural Networks: Methodology and Applications. illustrated ed. Birkhäuser; 2004.

[36] Egmont-Petersen M, De Ridder D, Handels H. Image Processing With Neural Networks - a Review. Pattern Recognition. 2002;35(10):2279-2301.

[37] Endo G, Janoyan V. Santa Monica Freeway Smart Corridor Project. Compendium of Technical Papers. Anaheim, California: Institute of Transportation Engineers; 1991.

[38] Faulkner X. Usability Engineering. New York: PALGRAVE, St. Martin's Press; 2000.

[39] Fellendorf M. VISSIM: A Microscopic Simulation Tool to Evaluate Actuated Signal Control Including Bus Priority. In: the 64th ITE Annual Meeting; 1994; Dalla. p. 1-9.

[40] Freilberger P, Mcneill D. Fuzzy Logic. Simon & Schuster; 1993.

[41] Fuzzy_Logic. The Columbia Encyclopedia, Sixth Edition. [Internet]. 2008 [cited 2009 July 04]. Available from: http://www.encyclopedia.com /doc/1E1-fuzzylogi.html.

[42] Ghezelayagh H, Lee K. Application of Self-Organized Neuro-Fuzzy Identifier in Intelligent Predictive Control of Power Plant. International Journal of Engineering Intelligent Systems for Electrical Engineering and Communications. 2005;13(2):113-118.

[43] Ghosh B, Basu B, O’Mahony M. Bayesian Time-Series Model for Short-Term Traffic Flow Forecasting. Journal of Transportation Engineering. 2007;133(3):180-189.

[44] Goldberg D. Genetic Algorithms is Search, Optimization and Machine Learning. Addion Wesley; 1989.

[45] Goldberg D, Deb K. A comparative analysis of selection schemes used in genetic algorithms. Foundations of Genetic Algorithms. 1991;1:69–93.

[46] Gonzblez A, Perez R. SLAVE: A Genetic Learning System Based on an Iterative Approach. IEEE Transactions on Fuzzy Systems. 1999;7(2):176-191.

[47] Gould J. How to Design Usable Systems. In: Baecker B, Grudin J, Buxton W, Greenberg S, editors. Human-Computer Interaction: Toward the Year 2000. San Francisco: Morgan Kaufmann Publishers Inc; 1995. p. 93-121.

[48] Grasso B, Ward M, Hall G, Perez C, Eiger A. ATMS and Wide-Area Traffic Management. In: The Annual Meeting of the ITS America; 1995. p. 543-553.

[49] Guan H, Ma W, Meng Y. Traffic Flow Prediction Based on Hierarchical Genetic Optimized Algorithm. In: the 3rd International Conference on Innovative Computing Information and Control, ICICIC '08.; 2008; Dalian. p. 121-127.

[50] Halgamuge S, Glesner M. Neural Networks in Designing Fuzzy Systems for Real World Applications. Fuzzy Sets and Systems. 1994;65(1):1-12.

[51] Haupt R, Haupt S. Practical Genetic Algorithms. John Wiley & Sons, Inc; 1998.

[52] Hauptmann W, Heesche K. A Neural Net Topology for Bidirectional Fuzzy-Neuro Transformation. In: IEEE International Conference on Fuzzy Systems (FUZZ-IEEE/IFES); 1995; Yokohama. p. 1511-1518.

[53] Hawkins D, Olwell D. Cumulative Sum Charts and Charting for Quality Improvement. New York: Springer; 1997.

[54] Hayashi Y, Buckley J. Approximations between Fuzzy Expert Systems and Neural Networks. International Journal of Approximate Reasoning. 1994;10:63-73.

[55] Hayashi I, Nomura H, Yamasaki H, Wakami N. Construction of fuzzy inference rules by NDF and NDFL. International Journal of Approximate Reasoning. 1992;6(2):241-266.

[56] Hecht-Nielsen R. Neurocomputing. Addison-Wesley; 1990.

[57] Hegyi A, De Schutter B, Hoogendoorn S, Babuska R, Van Zuylen H, Schuurman H. A Fuzzy Decision Support System for Traffic Control Centers. In: Proceedings IEEE Intelligent Transportation Systems; 2001; Oakland, California. p. 358-363.

[58] Helbing D. Gas-Kinetic Derivation of Navier–Stokes-Like Traffic Equations. Physical Review E. 1996;53(3):2366--2381.

[59] Helbing D. Verkehrsdynamik: Neue Physikalische Modellierungskonzepte. Springer; 1997.

[60] Helbing D, Hennecke A, Shvetsov V, Treiber M. MASTER:Macroscopic Traffic Simulation Based on a Gas-Kinetic, Non-local Traffic Model. Transportation Research B. 2001 February;35(2):183-211.

[61] Henry J, Farges J, Gallego J. Neuro-Fuzzy Techniques for Traffic Control.

Control Engineering Practice. 1998;6(6):755-761.

[62] Hernandez J, Ossowski S, Garcia-Serrano A. Multiagent Architectures for Intelligent Traffic Management Systems. Transportation Research Part C: Emerging Technologies. 2002;10(5-6):473-506.

[63] Heung T, Ho T. Hierarchical Fuzzy Logic Traffic Control at a Road Junction Using Genetic Algorithms. In: Proceedings IEEE International Conference on Fuzzy Systems; 1998; Anchorage, AK, USA. p. 1170-1175.

[64] Hoffmann F. Combining Boosting and Evolutionary Algorithms for Learning of Fuzzy Classification Rules. Fuzzy Sets and Systems. 2004;141(1):47-58.

[65] Holland J. Adaptation in Natural and Artificial Systems. Ann Arbor MI: University of Michigan Press. 1975.

[66] Homaifar A, McCormick E. Simultaneous Design of Membership Functions and Rule Sets for Fuzzy Controller Using Genetic Algorithms. IEEE Transactions on Fuzzy Systems. 1995;3(2):129-139.

[67] Hoogendoorn S, Bovy P. Continuum Modelling of Multiclass Traffic Flow. Transportation Research Part B. 2000;34:123-146.

[68] Hoogendoorn S, Bovy P. State-of-the-art of Vehicular Traffic Flow Modelling. Journal of Systems Control Engineer. 2001;215(14):283-303.

[69] Huang L, Huang K, Huang H, Jaw B. Applying Fuzzy Neural Network in Human Resource Selection System. In: IEEE Annual Meeting of the Fuzzy Information, NAFIPS'04.; 2004; Banff,Canada. p. 169-174.

[70] Hu Y, Thomas P, Stonier R. Traffic Signal Control Using Fuzzy Logic and Evolutionary Algorithms. In: IEEE Congress on Evolutionary Computation.; 2007; Singapore. p. 1785-1792.

[71] Ikeda H, Kaneko Y, Matsuo T, Tsuji K. Abnormal Incident Detection System Employing Image Processing Technology. In: IEEE/IEEJ/JSAI International Conference on Intelligent Transportation Systems; 1999; Tokyo, JAPAN. p. 748-752.

[72] Institute-of-Transportation-Engineers. Highway Capacity Manual. 3rd ed. DC: Washington; 1994.

[73] Irwin G, Warwick K, Hunt K. Neural Network Applications in Control. Institution of Engineering and Technology; 1995.

[74] Isaksen L, Payne H. Suboptimal Control of Linear Systems by Augmentation with Application to Freeway Traffic Regulation. IEEE Transactions on Automatic Control. 1973;18(3):210-219.

[75] Ishibuchi H, Tanaka H, Okada H. Interpolation of Fuzzy If-Then Rules by Neural Networks. International Journal of Approximate Reasoning. 1994;10(1):3–27.

[76] Ishibuchi H, Yamamoto T. Fuzzy Rule Selection by Multi-Objective Genetic Local Search Algorithms and Rule Evaluation Measures in Data Mining. Fuzzy Sets and Systems. 2004;141(1):59-88.

[77] Jang J. ANFIS: Adaptive-Network-Based Fuzzy Inference System. IEEE

Transaction on Fuzzy System, Man and Cybernetics. 1993;23(3):665–685.

[78] Jang J, Sun C, Mizutani E. Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence. Prentice Hall, Englewood Cliffs, NJ; 1997.

[79] Jiang G, Zhang R. Travel-Time Prediction for Urban Arterial Road: a Case on China. In: The IEEE International Vehicle Electronics Conference, IVEC01; 2001; Tottori, Japan. p. 255-260.

[80] Jin X, Cheu R, Srinivasan D. Development and Adaptation of Constructive Probabilistic Neural Network in Freeway Incident Detection. Transportation Research Part C: Emerging Technologies. 2002 April;10(2):121-147.

[81] Johnson P. Human-Computer Interaction. London: Mcgraw-Hill Book Company; 1992.

[82] Kachroo P, Ozbay K. Feedback Ramp Metering in Intelligent Transportation Systems. 1st ed. New York: Kluwer Academic/ Plenum Publishers; 2003.

[83] Kaedi M, Movahhedinia N, Jamshidi K. Traffic Signal Timing Using Two-Dimensional Correlation, Neuro-Fuzzy and Queuing based Neural Networks. Neural Computing & Applications. 2007 March;17(2):193-200.

[84] Kalyanmoy D. Optimization for Engineering Design: Algorithms and Examples. Prentice-Hall; 2004.

[85] Kamruzzaman J, Begg R, Sarker R, editors. Artificial Neural Networks in Finance and Manufacturing. illustrated ed. Idea Group Inc (IGI); 2006.

[86] Karr C. Genetic algorithms for fuzzy controllers. AI Expert. 1991;6(2):26-33.

[87] Kasabov N, Kim J, Watts M, Gray A. FuNN/2-A Fuzzy Neural Network Architecture for Adaptive Learning and Knowledge Acquisition. Information Sciences. 1997;101(3):155–175.

[88] Kell J, Fullerton I, Milton K. Traffic Detector Handbook. 2nd ed. Federal Highway Administration; 1990.

[89] Khan S, Ritchie S. Statistical and Neural Classifiers to Detect Traffic Operational Problems on Urban Arterials. Transportation Research Part C: Emerging Technologies. 1998;6(5):291-314.

[90] Kim J, Kasabov N. HyFIS: Adaptive Neuro-Fuzzy Inference Systems and Their Application to Nonlinear Dynamical Systems. Neural Networks. 1999;12(9):1301-1319.

[91] Kirschfink H, Hernández J, Boero M. Intelligent Traffic Management Models. In: European Symposium on Intelligent Techniques; 2000; Aachen, Germany. p. 36-45.

[92] Klein L, Gibson D, Mills M. Traffic Detector Handbook. 3rd ed. Federal Highway Administration; 2006.

[93] Klein L, Kelley M. Detection Technology for IVHS. Federal Highway Administration; Available through the National Technical Information Service; 1996.

[94] Klügl F, Bazzan A, Ossowski S, editors. Applications of Agent Technology in Traffic and Transportation. 2nd ed. Basel: Birkhäuser; 2005.

[95] Kohonen T. Self-Organization and Associative Memory. 2nd ed. Berlin: Springer-Verlag; 1988.

[96] Kosko B. Fuzzy Thinking: The New Science of Fuzzy Logic. 1st ed. New York: Flamingo; 1994.

[97] Kosko B. Unsupervised Learning in Noise. IEEE Transactions on Neural Networks. 1990 Mar.;1(1):44-57.

[98] Krause B, Von Altrock C, Limper K, Schafers W. A Neuro-Fuzzy Adaptive Control Strategy for Refuse Incineration Plants. Fuzzy Sets and Systems. 1994;63(3):329-338.

[99] Lee Y, Hwang C, Shih Y. A Combined Approach to Fuzzy Model Identification. IEEE Transactions on System, Man and Cybernetics. 1994;24(5):736-744.

[100] Lee S, Krammes R, Yen J. Fuzzy-Logic-based Incident Detection for Signalized Diamond Interchanges. Transportation Research Part C: Emerging Technologies. 1998;6(5):359-377.

[101] Lee S, Lyou J. Neurofuzzy Modeling of Manual Control System with a Human Operator. In: International Conference on Control, Automation and Systems, ICCAS'07; 2007; Seoul. p. 767-770.

[102] Liao T, Celmins A, Hammell R. A fuzzy C-means Variant for the Generation

of Fuzzu Term Sets. Fuzzy Sets and Systems. 2003;135(2):241-257.

[103] Lin Y, Cunningham III G. A New Approach to Fuzzy-Neural System Modelling. IEEE Transactions on Fuzzy Systems. 1995;3(2):190-198.

[104] Lin C, Lee C. Neural-Network-Based Fuzzy Logic Control and Decision System. IEEE Transactions on Computers. 1991;40(12):1320-1336.

[105] Lin C, Lin C. An ART-based Fuzzy Adaptive Learning Control Network. IEEE Transaction on Fuzzy System. 1997;5:477–496.

[106] Li T, Zhao D, Yi J. Adaptive Dynamic Neuro-Fuzzy System for Traffic Signal Control. In: IEEE International Joint Conference on Neural Networks, IJCNN08.; 2008; Hong Kong. p. 1840-1846.

[107] Luo X, Hou W, Li Y, Wang Z. A Fuzzy Neural Network Model for Predicting Clothing Thermal Comfort. Computers and Mathematics with Applications. 2007;53(12):1840-1846.

[108] Malamas E, Petrakis E, Zervakis M, Petit L, Legat J. A Survey on Industrial Vision Systems, Applications and Tools. Image and Vision Computing. 2003;21(2):171-188.

[109] Malej A, Brodnik A. Fuzzy Urban Traffic Signal Control - an Overview. Elektrotehniski Vestnik. 2007;74(5):291-296.

[110] Mamdani E, Assilian S. An experiment in linguistic synthesis with a fuzzy controller. International Journal of Man-Machine Studies. 1975;7(1):1-13.

[111] Mazzetti C, Mascioli F, Baldini F, Panella M, Risica R, Bartnikas R. Partial Discharge Pattern Recognition by Neuro-Fuzzy Networks in Heat-Shrinkable Joints And Terminations of XLPE Insulated Distribution Cables. IEEE Transactions on Power Delivery. 2006;21(3):1035-1044.

[112] Meignan D, Simonina O, Koukama A. Simulation and Evaluation of Urban Bus-Networks Using a Multiagent Approach. Simulation Modelling Practice and Theory. 2007 July;15(6):659-671.

[113] Messmer A, Papageorgiou M. METANET: A macroscopic simulation program for motorway networks. Traffic Engineering and Control. 1990;31(9):466-470.

[114] Middelham F. State of Practice in Dynamic Traffic Management in The Netherlands. In: the 10th IFAC Symposium on Control in Transportation Systems (CTS03); 2003; Tokyo, Japan.

[115] Middleton D, Gopalakrishna D, Raman M. Advances in Traffic Data Collection and Management. White paper. Washington: Texas Transportation Institute, Cambridge Systematics, Inc.; 2003. BAT-02-006.

[116] Mitchell M. An Introduction to Genetic Algorithms. Cambridge: MIT Press; 1996.

[117] Mitra S, Hayashi Y. Neuro-Fuzzy Rule Generation: Survey in Soft Computing Framework. IEEE Transaction on Neural Networks. 2000 May;11(3):748-768.

[118] Negnevitsky M. Artificial Intelligence: A Guide to Intelligent Systems. 2nd ed. Boston: Addison-Wesley Longman Publishing Co.; 2004.

[119] Nguyen M, Shi D, Quek C, Ng G. Traffic Prediction Using Ying-Yang Fuzzy Cerebellar Model Articulation Controller. In: the 18th International Conference on Pattern Recognition ICPR06.; 2006; Hong Kong. p. 258-261.

[120] Nie J. Constructing Fuzzy Model by Self-Organising Counterpropagation Network. IEEE Transactions on System, Man and Cybernetics. 1995;25(6):963-970.

[121] Nielsen J. Usability 101: Introduction to Usability. [Internet]. 2003 [cited 2009 Dec 24]. Available from: http://www.useit.com/alertbox/ 20030825.html.

[122] Nogduy D, Hoogendoorn S. An Automated Calibration Procedure for Macroscopic Traffic Flow Models. In: the 10th IFAC Symposium on Control in Transportation Systems (CTS03); 2003; Tokyo, Japan. p. 295–300.

[123] Nomura H, Hayashi I, Wakami N. A self-tuning method of fuzzy reasoning by genetic algorithm. In: IEEE International Conference on Fuzzy Systems and Intelligent Control; 1992; Louisville, USA. p. 236–245.

[124] Ossowski S. Co-Ordination in Artificial Agent Societies: Social Structures and Its Implications for Autonomous Problem-Solving Agents. Vol 1535. Springer Verlag; 1999.

[125] Papageorgiou M. Dynamic Modeling, Assignment, and Route Guidance in

Traffic Networks. Transportation Research Part B. 1990;24B(6):471-495.

[126] Papageorgiou M, Kotsialos A. Freeway Ramp Metering: An Overview. IEEE Transactions on Intelligent Transportation Systems. 2002;3(4):271-280.

[127] Park B. Hybrid Neuro-Fuzzy Application in Short-Term Freeway Traffic Volume Forecasting. Transportation Research Record: Journal of the Transportation Research Board. 2002;1802(361):190-196.

[128] Parkany E, Chi X. Use of Driver-Based Data for Incident Detection. In: Proceeding the 7th International Conference in Applications of Advanced Technologies in Transportation; 2002; Cambridge. p. 143-150.

[129] Pedrycz W. An Identification Algorithm in Fuzzy Relational Systems. Fuzzy Sets and Systems. 1984;13:153-167.

[130] Pedrycz W, Patrick C, Rocha A. Distributed Fuzzy System Modelling. IEEE Transactions on System, Man and Cybernetics. 1995;25(5):769-780.

[131] Qiao F, Yang H. Intelligent Simulation and Prediction of Traffic Flow Dispersion. Transportation Research Part B: Methodological. 2001;35(9):843-863.

[132] Quadstone. Paramics v4.0 System Overview. Technical Report. Edinburgh, Scotland: Quadstone Limited; 2002.

[133] Quek C, Pasquier M, Lim B. A Novel Self-Organizing Fuzzy Rule-Based System for Modelling Traffic Flow Behaviour. Expert Systems With Applications. 2009;36(10):12167-12178.

[134] Quek C, Pasquier M, Lim B. POP-TRAFFIC: A Novel Fuzzy Neural Approach to Road Traffic Analysis and Prediction. IEEE Transactions on Intelligent Transportation Systems. 2006 June;7(2):133-146.

[135] Quek C, Tung W. Falcon: Fuzzy Neural Control and Decision Systems Using FKP and PFKP Dustering Algorithms. IEEE Transactions on Systems, Man and Cybernetics, Part B. 2004 Feb.;34(1):686-694.

[136] Quek C, Zhou R. The POP Learning Algorithms: Reducing Work in Identifying Fuzzy Rules. Neural Networks. 2001;14(10):1431-1445.

[137] Quek C, Zhou R. POPFNN: A Pseudo Outer-Product Based Fuzzy Neural Network. Neural Network. 1996;9(9):1569-1581.

[138] Quek C, Zhou R. POPFNN-AAR (S): a Pseudo Outer-Product based Fuzzy Neural Network. IEEE Transactions on Systems, Man and Cybernetics, Part B. 1999;29(6):859-870.

[139] Remde S, Cowling P, Dahal K, Colledge N. Exact/Heuristic Hybrids Using rVNS and Hyperheuristics for Workforce Scheduling. Lecture Notes in Computer Science. 2007 April;4446:188-197.

[140] Ritchie SG. A Knowledge-Based Decision Support Architecture for Advanced Traffic Management. Transportation Research Part A: General. 1990;24(1):27-37.

[141] Sadek A, Smith B, Demetsky M. A Prototype Case-Based Reasoning System for Real-Time Freeway Traffic Routing. Transportation Research Part C:

Emerging Technologies. 2001 October;9(5):353-380.

[142] Sen A, Thakuriah P, Zhu X, Karr A. Variances of Link Travel Time Estimates: Implications for Optimal Routes. International Transactions in Operational Research. 1999;6(75):75-87.

[143] Sentinella D, Hardman E. Review of Motorway Speed Control Systems in Europe. Technical Report. unpublished; 1996. 056/96.

[144] Setiono R, Leow W. FERNN: An Algorithm for Fast Extraction of Rules from Neural Networks. Applied Intelligence. 2000;12(1):15-25.

[145] Shann J, Fu H. A Fuzzy Neural Network for Rule Acquiring on Fuzzy Control Systems. Fuzzy Sets and Systems. 1995;71(3):345-357.

[146] Specht D. Probabilistic Neural Networks. Neural networks. 1990;3(1):109-118.

[147] Srinivasan D, Cheu R, Poh Y. Hybrid Fuzzy Logic-Genetic Algorithm Technique for Automated Detection of Traffic Incidents on Freeways. In: Proceedings IEEE Intelligent Transportation Systems; 2001; Oakland, CA, USA. p. 352-357.

[148] Srinivasan D, Cheu R, Poh Y, Ng A. Development of an Intelligent Technique for Traffic Network Incident Detection. Engineering Applications of Artificial Intelligence. 2000 June;13(1):311-322.

[149] Srinivasan D, Jin X, Cheu R. Evaluation of Adaptive Neural Network Models for Freeway Incident Detection. IEEE Transactions on Intelligent

Transportation Systems. 2004;5(1):1-11.

[150] Stathacopoulou R, Magoulas G, Grigoriadou M, Samarakou M. Neuro-fuzzy Knowledge Processing in Intelligent Learning Environments for Improved Student Diagnosis. Information Sciences. 2005;170(2):273-307.

[151] Stone P, Veloso M. Multiagent Systems: A Survey from a Machine Learning Perspective. Autonomous Robots. 2000;8(3):345-383.

[152] Sugeno M, Yasukawa T. A Fuzzy-Logic-Based Approach to Qualitative Modelling. IEEE Transactions on Fuzzy Systems. 1993;1(1):7-31.

[153] Sugeno M, Yasukawa T. Linguistic Modelling Based on Numerical Data. In: Proceedings IFSA ‘91, Brussels: Computer, Management & Systems Science.; 1991; Brussels. p. 264-267.

[154] Surmann H, Kanstein A, Goser K. Self-Organizing and Genetic Algorithms for an Automatic Design of Fuzzy Control and Decision Systems. In: the First European Congress on Fuzzy and Intelligent Technologies, EUFIT' 93; 1993; Aachen. p. 1097-1104.

[155] Su Z, Shuguang Z, Yu L, Li'ai G. New Temperature Idity Fuzzy Neural Neural Network Controller Based On Improved Genetic Algorithm. In: the 8th International Conference on Electronic Measurement and Instruments ICEMI'07; 2007; Xi'an. p. 2590-2594.

[156] Sutton R, editor. Reinforcement Learning. Springer; 1992.

[157] Takagi T, M. Sugeno. Fuzzy Identification of Systems and Its Application to

Modeling and Control. IEEE Transaction on Fuzzy System, Man and Cybernetics. 1985;SMC-15:116–132.

[158] Takagi T, Sugeno M. Derivation of Fuzzy Control Rules From Human Operator’s Control Actions. In: Proceedings the IFAC Symposium on Fuzzy Information, Knowledge Representation and Decision Analysis; 1983; Marseille, France. p. 55-60.

[159] Tang S, Gao H. Traffic-Incident Detection-Algorithm Based on Nonparametric Regression. IEEE Transactions on Intelligent Transportation Systems. 2005;6(1):38-42.

[160] Tong R. The Evaluation of Fuzzy Models Derived from Experimental Data. Fuzzy Sets and Systems. 1980;4(1):1-12.

[161] Trabia M, Kaseko M, Ande M. A Two-Stage Fuzzy Llogic Controller for Traffic Signals. Transportation Research Part C: Emerging Technologies. 1999;7(6):353-367.

[162] Tung W, Quek C. GenSoFNN: A Generic Self Organizing Fuzzy Neural Network. IEEE Transactions on Neural Networks. 2002;13(5):1075–1086.

[163] Van Aerde M, Yagar S. Dynamic integrated freeway/traffic signal networks: problems and proposed solutions. Transportation research. Part A: general. 1988;22(6):435-443.

[164] Van Den Berg M, Hegyi A, De Schutter B, Hellendoom J. A Macroscopic Traffic Flow Model for Integrated Control of Freeway and Urban Traffic

Networks. In: the 42nd IEEE Conference on Decision and Control; 2003; Maui, Hawaii. p. 2774-2779.

[165] Van Katwijk R, Van Koningsbruggen P, De Schutter B, Hellendoorn J. Test Bed for Multiagent Control Systems in Road Traffic Management. Transportation Research Record. 2005;1910(1):108-115.

[166] Vas P. Artificial-Intelligence-Based Electrical Machines and Drives: Application of Fuzzy, Neural, Fuzzy-neural, and Genetic-Algorithm-based Techniques (Monographs in Electrical and Electronic Engineering). illustrated ed. New York: Oxford University Press, Inc; 1999.

[167] Wahle J, Neubert L, Esser J, Schreckenberg M. A Cellular Automaton Traffic Flow Model for Online Simulation of Traffic. Parallel Computing. 2001;27(5):719-735.

[168] Wang W, Cheng C, Leu Y. An Online GA-Based Output-Feedback Direct Adaptive Fuzzy-Neural Controller for Uncertain Nonlinear Systems. IEEE Transactions on Systems, Man and Cybernetics, Part B. 2004;34(1):334-345.

[169] Wang W, Lee T, Hsu C, Li Y. GA-Based Learning of BMF Fuzzy-Neural Network. In: IEEE International Conference on Fuzzy Systems; 2002; Honolulu, HI, USA. p. 1234-1239.

[170] Wang L, Mendel J. Generating Fuzzy Rules by Learning from Examples. IEEE Transactions on System, Man and Cybernetics. 1992;22(6):1414–1427.

[171] Weil R, Garcia-Ortiz A, Wootton J. Detection of Traffic Anomalies Using

Fuzzy Logic Based Techniques. In: Proceedings IEEE International Conference on Fuzzy Systems; 1998; Anchorage, AK, USA. p. 1176-1181.

[172] Wei W, Zhang Y, Mbede J, Zhang Z, Song J. Traffic Signal Control Using Fuzzy Logic and MOGA. In: IEEE International Conference on Systems, Man, and Cybernetics; 2001; Tucson, Arizona. p. 1335-1340.

[173] Wen H, Yang Z, Jiang G, Shao C. A New Algorithm of Incident Detection on Freeways. In: IEEE International Vehicle Electronics Conference, IVEC01; 2001; Tottori, Japan. p. 197-202.

[174] Williams B. Highway Control. IEE Review. 1996;42(5):191-194.

[175] Williams B, Guin A. Traffic Management Center Use of Incident Detection Algorithms: Findings of a Nationwide Survey. IEEE Transactions on Intelligent Transportation Systems. 2007 June;8(2):351-358.

[176] Wooldridge M. An Introduction to Multiagent Systems. 1st ed. Jon Wiley & Sons; 2002.

[177] Wooldridge M, Jennings N. Intelligent Agents: Theory and Practice. Knowledge Engineering Review. 1995;10(2):115-152.

[178] WSDOT. WSDOT. [Internet]. 2009 [cited 2009 May 28]. Available from: http://www.wsdot.wa.gov/Projects/I5/VariableSpeedSafety/arrowgantry.htm.

[179] Wu C, Liu G. A Genetic Approach for Simultaneous Design of Membership Functions and Fuzzy Control Rules. Journal of Intelligent and Robotic Systems. 2000 July;28(3):195-211.

[180] Xu H, Kwan C, Haynes L, Pryor J. Real-Time Adaptive On-Line Traffic Incident Detection. In: IEEE International Symposium on Intelligent Control; 1996; Dearborn, USA. p. 200-205.

[181] Xu C, Lehr M. Fuzzy Model Identification and Self-Learning for Dynamic Systems. IEEE Transactions on Systems, Man, and Cybernetics. 1987;17(4):683-689.

[182] Yager R. Modeling and Formulating Fuzzy Knowledge Bases Using Neural Networks. Neural Networks. 1994;7(8):1273-1283.

[183] Yager R, Filev D. Essentials of Fuzzy Modeling and Control. 1st ed. Wiley; 1994.

[184] Yin H, Wong S, Xu J, Wong C. Urban Traffic Flow Prediction Using a Fuzzy-Neural Approach. Transportation Research Part C: Emerging Technologies. 2002;10(2):85-98.

[185] Young T. M25 Speed Cameras Go Digital. [Internet]. 2008 [cited 2009 May 28]. Available from: http://www.businesscar.co.uk/story.asp?storycode =2575.

[186] Yu L, Yu L, Wang J, Qi Y, Wen H. Back-Propagation Neural Network for Traffic Incident Detection Based on Fusion of Loop Detector and Probe Vehicle Data. In: the 4th International Conference on Natural Computation, ICNC '08.; 2008; Jinan. p. 116-120.

[187] Zadeh L. Fuzzy Logic for the Management of Uncertainty. New York:

Wiley-Interscience; 1992.

[188] Zadeh L. Fuzzy Sets. Information and Control. 1965;8:338-353.

[189] Zang L, Jia L, Luo Y. An Intelligent Control Method for Urban Traffic Signal Based on Fuzzy Neural Network. In: he 6th World Congress on Intelligent Control and Automation, WCICA06.; 2006; Dalian. p. 3430-3434.

[190] Zeigler B. High Autonomy Systems: Concepts and Models. In: Proceedings AI, Simulation, and Planning in High Autonomy Systems; 1990; Tucson, USA. p. 2-7.

[191] Zhang H. Structural Properties of Solutions Arising from a Nonequilibrium Traffic Flow Theory. Transportation Research Part B. 2000;34:583-603.

[192] Zhang H, Ritchie S. Real-Time Decision-Support System for Freeway Management and Control. Journal of Computing in Civil Engineering. 1994;8(1):35-51.

[193] Zhang H, Ritchie S, Jayakrishnan R. Coordinated Traffic-Responsive Ramp Control via Nonlinear State Feedback. Transportation Research Part C: Emerging Technologies. 2001;9(5):337-352.

[194] Zhu A, Yang S. Neurofuzzy-Based Approach to Mobile Robot Navigation in Unknown Environments. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews. 2007;37(4):610-621.

Appendix A

Series J Data from Box and Jenkins

-0.109	53.8	0	53.6	0.178	53.5	0.339	53.5
0.373	53.4	0.441	53.1	0.461	52.7	0.348	52.4
0.127	52.2	-0.18	52	-0.588	52	-1.055	52.4
-1.421	53	-1.52	54	-1.302	54.9	-0.814	56
-0.475	56.8	-0.193	56.8	0.088	56.4	0.435	55.7
0.771	55	0.866	54.3	0.875	53.2	0.891	52.3
0.987	51.6	1.263	51.2	1.775	50.8	1.976	50.5
1.934	50	1.866	49.2	1.832	48.4	1.767	47.9
1.608	47.6	1.265	47.5	0.79	47.5	0.36	47.6
0.115	48.1	0.088	49	0.331	50	0.645	51.1
0.96	51.8	1.409	51.9	2.67	51.7	2.834	51.2
2.812	50	2.483	48.3	1.929	47	1.485	45.8
1.214	45.6	1.239	46	1.608	46.9	1.905	47.8
2.023	48.2	1.815	48.3	0.535	47.9	0.122	47.2
0.009	47.2	0.164	48.1	0.671	49.4	1.019	50.6
1.146	51.5	1.155	51.6	1.112	51.2	1.121	50.5
1.223	50.1	1.257	49.8	1.157	49.6	0.913	49.4
0.62	49.3	0.255	49.2	-0.28	49.3	-1.08	49.7
-1.551	50.3	-1.799	51.3	-1.825	52.8	-1.456	54.4
-0.944	56	-0.57	56.9	-0.431	57.5	-0.577	57.3
-0.96	56.6	-1.616	56	-1.875	55.4	-1.891	55.4
-1.746	56.4	-1.474	57.2	-1.201	58	-0.927	58.4
-0.524	58.4	0.04	58.1	0.788	57.7	0.943	57
0.93	56	1.006	54.7	1.137	53.2	1.198	52.1
1.054	51.6	0.595	51	-0.08	50.5	-0.314	50.4
-0.288	51	-0.153	51.8	-0.109	52.4	-0.187	53

-0.255	53.4	-0.229	53.6	-0.007	53.7	0.254	53.8
0.33	53.8	0.102	53.8	-0.423	53.3	-1.139	53
-2.275	52.9	-2.594	53.4	-2.716	54.6	-2.51	56.4
-1.79	58	-1.346	59.4	-1.081	60.2	-0.91	60
-0.876	59.4	-0.885	58.4	-0.8	57.6	-0.544	56.9
-0.416	56.4	-0.271	56	0	55.7	0.403	55.3
0.841	55	1.285	54.4	1.607	53.7	1.746	52.8
1.683	51.6	1.485	50.6	0.993	49.4	0.648	48.8
0.577	48.5	0.577	48.7	0.632	49.2	0.747	49.8
0.9	50.4	0.993	50.7	0.968	50.9	0.79	50.7
0.399	50.5	-0.161	50.4	-0.553	50.2	-0.603	50.4
-0.424	51.2	-0.194	52.3	-0.049	53.2	0.06	53.9
0.161	54.1	0.301	54	0.517	53.6	0.566	53.2
0.56	53	0.573	52.8	0.592	52.3	0.671	51.9
0.933	51.6	1.337	51.6	1.46	51.4	1.353	51.2
0.772	50.7	0.218	50	-0.237	49.4	-0.714	49.3
-1.099	49.7	-1.269	50.6	-1.175	51.8	-0.676	53
0.033	54	0.556	55.3	0.643	55.9	0.484	55.9
0.109	54.6	-0.31	53.5	-0.697	52.4	-1.047	52.1
-1.218	52.3	-1.183	53	-0.873	53.8	-0.336	54.6
0.063	55.4	0.084	55.9	0	55.9	0.001	55.2
0.209	54.4	0.556	53.7	0.782	53.6	0.858	53.6
0.918	53.2	0.862	52.5	0.416	52	-0.336	51.4
-0.959	51	-1.813	50.9	-2.378	52.4	-2.499	53.5
-2.473	55.6	-2.33	58	-2.053	59.5	-1.739	60
-1.261	60.4	-0.569	60.5	-0.137	60.2	-0.024	59.7
-0.05	59	-0.135	57.6	-0.276	56.4	-0.534	55.2
-0.871	54.5	-1.243	54.1	-1.439	54.1	-1.422	54.4
-1.175	55.5	-0.813	56.2	-0.634	57	-0.582	57.3
-0.625	57.4	-0.713	57	-0.848	56.4	-1.039	55.9
-1.346	55.5	-1.628	55.3	-1.619	55.2	-1.149	55.4
-0.488	56	-0.16	56.5	-0.007	57.1	-0.092	57.3
-0.62	56.8	-1.086	55.6	-1.525	55	-1.858	54.1
-2.029	54.3	-2.024	55.3	-1.961	56.4	-1.952	57.2
-1.794	57.8	-1.302	58.3	-1.03	58.6	-0.918	58.8
-0.798	58.8	-0.867	58.6	-1.047	58	-1.123	57.4
-0.876	57	-0.395	56.4	0.185	56.3	0.662	56.4
0.709	56.4	0.605	56	0.501	55.2	0.603	54
0.943	53	1.223	52	1.249	51.6	0.824	51.6
0.102	51.1	0.025	50.4	0.382	50	0.922	50

1.032	52	0.866	54	0.527	55.1	0.093	54.5
-0.458	52.8	-0.748	51.4	-0.947	50.8	-1.029	51.2
-0.928	52	-0.645	52.8	-0.424	53.8	-0.276	54.5
-0.158	54.9	-0.033	54.9	0.102	54.8	0.251	54.4
0.28	53.7	0	53.3	-0.493	52.8	-0.759	52.6
-0.824	52.6	-0.74	53	-0.528	54.3	-0.204	56
0.034	57	0.204	58	0.253	58.6	0.195	58.5
0.131	58.3	0.017	57.8	-0.182	57.3	-0.262	57